PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s): Denis Proulx, et al.

Title: METHOD AND SYSTEM FOR IP LINK MANAGEMENT

App. No.: 10/027,821 Filed: 12-19-2001

Examiner: Ke, Peng Group Art Unit: 2174

Atty. Docket No.: 1400.1374890

Mail Stop Appeal Brief - Patents
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Dear Sir:

In response to the notice of appeal filed September 18, 2009, and the notification of non-compliant appeal brief mailed May 11, 2010, Appellant submits the following:

This appeal is taken under 35 U.S.C. § 134. A final Office action rejecting claims 1-18 has a mail date of March 18, 2009. A notice of appeal and a petition for extension of time were received at the United States Patent and Trademark Office on September 24, 2009. An appeal brief and a petition for extension of time were mailed April 26, 2010. A notification of non-compliant appeal brief has a mail date of May 11, 2010. This amended appeal brief is being mailed June 11, 2010.

On June 10, 2008, the Patent and Trademark Office published a final rule at 73 FR 32938 et seq. amending 37 CFR 41.37 to impose new requirements on appeal briefs. In the notification of non-compliant appeal brief mailed May 11, 2010, the Examiner states, "...the Office will now no longer accept appeal briefs in the new format." Thus, Applicant submits this amended appeal brief. Applicant respectfully requests the Board consider this amended appeal brief or the previously filed appeal brief, as appropriate.

REAL PARTY IN INTEREST

As presently advised, Alcatel-Lucent Canada Inc. is the real party in interest in this appeal by virtue of an executed Assignment of the entire interest from the named Inventor(s), Denis Proulx, Chuong Ngoc Ngo, Attaullah Zabihi, David Wing-Chung Chan and Felix Katz, to Alcatel Canada Inc., recorded in the United States Patent and Trademark Office on 05-09-2002 at Reel 012876, Frame 0474, followed by a Certificate of Amalgamation from Alcatel Canada Inc. to Alcatel-Lucent Canada Inc. dated January 1, 2007. Appellant encloses copies of the above-referenced Assignment and Certificate of Amalgamation.

RELATED APPEALS AND INTERFERENCES

As presently advised, there are no other prior or pending appeals, interferences, or judicial proceedings known to Appellant, the Appellant's legal representative, or Assignee which may be related to, directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal.

STATUS OF CLAIMS

Claims 1-18 are pending in the present application. Claims 1-18 are finally rejected, the rejection of which is being appealed.

STATUS OF AMENDMENTS

Appellant has not amended the specification, drawings, or claims subsequent to final rejection

SUMMARY OF CLAIMED SUBJECT MATTER

Independent claim 1 recites a network administration method for provisioning logical configuration links for at least two network devices through a dedicated graphical user interface form **{Page 31, line 20, through page 32, line 13; Fig. 5}**, the method comprising:

selecting a network device having at least one network interface through the dedicated graphical user interface form **{Page 31, line 23, through page 32, line 1; Fig. 5, step 501}**;

determining local interface and next neighbor information for the network device **{Page 32, lines 1 and 2; Fig. 5, step 502}**;

determining whether the local interface and next neighbor information is associated with a logical configuration link stored among a plurality of logical configuration links in a logical link database **{Page 32, lines 2 through 5; Fig. 5, step 503}**;

creating a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database **{Page 32, lines 6 and 7; Fig. 5, step 504}**;

storing the new logical configuration link in the logical link database **{Page 32, lines 7 and 8; Fig. 5, step 505}**;

validating the new logical configuration link **{Page 32, lines 8 and 9; Fig. 5, step 506}**;

sending the new logical configuration link to the network device **{Page 32, lines 9 and 10; Fig. 5, step 507}**; and

displaying a graphical representation of the new logical configuration link on a display device **{Page 32, lines 10 and 11; Fig. 5, step 508}**.

Dependent claim 2 depends from claim 1 and recites subject matter wherein the step of creating a new logical configuration link further comprises **{Page 32, lines 14 through 16; Fig. 6}** the steps of:

selecting a link type **{Page 32, lines 16 and 17; Fig. 6, step 601}**;

selecting a link numbering type for the new logical configuration link **{Page 32, lines 18 and 19; Fig. 6, step 603}**;

selecting a link application for the new logical configuration link **{Page 32, lines 21 and 22; Fig. 6, step 605}**;

selecting a sub layer interface type for the new logical configuration link **{Page 32, lines 24 and 25; Fig. 6, step 607}**;

creating a first endpoint for the new logical configuration link **{Page 33, line 1; Fig. 6, step 609}**; and

creating a second endpoint for the new logical configuration link **{Page 33, line 2; Fig. 6, step 610}**.

Dependent claim 3 depends from claim 2 and recites subject matter wherein the step of selecting the link type further comprises **{Page 32, line 17}** the step of:

selecting the link type from among a group consisting of: point-to-point, point-to-IP, and point-to-subnet **{Page 32, lines 17 and 18; Fig. 6, step 602}**.

Dependent claim 4 depends from claim 2 and recites subject matter wherein the step of selecting a link numbering type further comprises **{Page 32, lines 19 and 20}** the step of:

selecting the link numbering type from a group consisting of: a numbered type and an unnumbered type **{Page 32, lines 20 and 21; Fig. 6, step 604}**.

Dependent claim 5 depends from claim 2 and recites subject matter wherein the step of selecting a link application further comprises **{Page 32, line 22}** the step of:

selecting the link application from a group consisting of: Internet Protocol Forwarding, Multi-Protocol Label Switching and Internet Protocol Forwarding, and Multi-Protocol Label Switching **{Page 32, lines 22 through 24; Fig. 6, step 606}**.

Dependent claim 6 depends from claim 2 and recites subject matter wherein the step of selecting a sub layer interface type further comprises **{Page 32, line 25}** the step of:

selecting the sub layer interface type from a group consisting of: Packet Over Sonet, Asynchronous Transfer Mode, and GigEthernet **{Page 32, line 25, through page 33, line 1; Fig. 6, step 608}**.

Dependent claim 7 depends from claim 1 and recites subject matter further comprising the step of:

modifying a logical configuration link in the logical link database **{Page 32, line 12; Fig. 5, step 509}**.

Dependent claim 8 depends from claim 1 and recites subject matter further comprising the step of:

deleting a logical configuration link in the logical link database **{Page 32, lines 12 and 13; Fig. 5, step 510}**.

Independent claim 9 recites apparatus for provisioning logical configuration links **{Page 33, line 3, through Page 35, line 5; Fig. 7}** comprising:

a logical link database for storing logical configuration links **{Page 33, line 13; Fig. 7, logical link database 705}**;

a processing system coupled to the logical link database for accessing the logical link database **{Page 33, lines 6, 7, 13, and 14; Fig. 7, processing system 704}**; and

a display device coupled to the processing system for displaying a graphical user interface form comprising a graphical representation of a logical configuration link **{Page 33, lines 4-6, 14, and 15; Fig. 7, display device 701}**.

Dependent claim 10 depends from claim 9 and recites subject matter wherein the display device provides an ability to select a network device having at least one network interface through the graphical user interface form **{Page 33, lines 14 and 15}**.

Dependent claim 11 depends from claim 9 and recites subject matter wherein the processing system determines local interface and next neighbor information for the network device **{Page 33, lines 16 and 17}**.

Dependent claim 12 depends from claim 11 and recites subject matter wherein the processing system determines whether the local interface and next neighbor information is associated with one of the logical configuration links stored in the logical link database **{Page 33, lines 18 and 19}**.

Dependent claim 13 depends from claim 12 and recites subject matter wherein the processing system creates a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links stored in the logical link database **{Page 33, lines 20 through 22}**.

Dependent claim 14 depends from claim 13 and recites subject matter wherein the processing system causes the new logical configuration link to be stored in the logical link database **{Page 33, lines 23 and 24}**.

Dependent claim 15 depends from claim 14 and recites subject matter wherein the processing system validates the new logical configuration link **{Page 33, line 24}**.

Dependent claim 16 depends from claim 15 and recites subject matter wherein the processing system causes the new logical configuration link to be sent to the network device **{Page 33, lines 25 and 26}**.

Dependent claim 17 depends from claim 1 and recites subject matter wherein creating the new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database occurs based on interfaces information entered by a user **{Page 12, line 2, through page 31, line 19; Figs. 3A-3C and 4}**.

Independent claim 18 recites a method comprising:

selecting a link type **{Page 13, lines 15, through page 14, line 23; Fig. 3A, step 303}**;

selecting a link numbering type **{Page 14, lines 24, through page 17, Table 2; Fig. 3A, step 313}**;

selecting a link application **{Page 17, lines 2 through 8; Fig. 3A, step 319}**;

selecting a sub layer interface type **{Page 17, line 9, through page 19, line 11; Fig. 3A, step 327}**;

creating a first endpoint **{Page 19, line 12, through page 21, line 8; Fig. 3B, step 339}**;

creating a second endpoint **{Page 21, line 9, through page 22, line 10; Fig. 3B, step 347}**;

populating form panels with the link type, the link numbering type, the link application, and the sub layer interface type **{Page 22, lines 11 through 16; Fig. 3C, step 355}**;

receiving user input of interfaces information **{Page 22, lines 17, through page 23, line 9; Fig. 3C, step 359}**;

validating the interfaces information **{Page 23, line 10, through page 24, Table 5; Fig. 3C, step 366}**;

creating a link in accordance with the interfaces information **{Page 23, line 10, through page 24, line 13; Fig. 3C, step 366}**; and

provisioning the link {Page 24, lines 2 through 13; Fig. 3C, step 370}.

In 69 F.R. 49960 at 49976, in the third column, lines 21-25, the following statement is set forth: "Whether the explanation is limited to a single drawing or embodiment or is extended to all drawings and embodiments is a decision appellant will need to make." Accordingly, while Appellant has provided examples above with respect to particular drawings, it should be understood that, for at least some claims, disclosure found in other portions of the specification and/or drawings may be useful toward obtaining a full appreciation of the scope of such claims.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The grounds of rejection to be reviewed on appeal are as follow:

Claim Rejections - 35 *USC* § *102*

Claims 1-4, and 7-18 are rejected under 35 U.S.C. 102(b) as being anticipated by Hansen US Patent 5,838,907.

Claim Rejections - 35 *USC* § *103*

Claim 5 is rejected under 35 U.S.C. 103(a) as being unpatentable over Hansen in view of Hansen US Patent 5,838,907 [sic] in view of Hardwick US Patent 5,550,816.

Claims 6 is rejected under 35 U.S.C. 103(a) as being unpatentable over Hansen in view of Hansen US Patent 5,838,907 [sic] in view of Chui US Patent 2002/0165978.

ARGUMENT

Rejection of Claims 1-4 and 7-18 under 35 U.S.C. 102(b) as being anticipated by Hansen US Patent 5,838,907

Appellant submits there exist clear errors in the Examiner's rejections and/or the Examiner's omissions of one or more essential elements needed for a prima facie rejection. Appellant submits the Examiner's "Response to Arguments" provides evidence that the Examiner has failed to consider the pending claims as required by the Manual of Patent Examining Procedure (MPEP) and prevailing case law. For anticipation under 35 U.S.C. § 102, a reference must teach <u>every aspect</u> of the claimed invention either explicitly or implicitly. Any feature not directly taught must be inherently present [emphasis added]. Appellant submits MPEP § 2131 provides: "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described in a single prior art reference. *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631 (Fed. Cir. 1987). 'The identical invention must be shown in as complete detail as contained in the...claim.' *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236 (Fed. Cir. 1989). The elements must be arranged as required by the claim." MPEP § 2141 sets forth the Graham inquiries for a rejection under 35 U.S.C. § 103. As Appellant describes in detail below, Appellant submits there exist clear errors in the Examiner's rejections and/or the Examiner's omissions of one or more aspects of a *prima facie* rejection.

Regarding claim 1, Appellant submits the cited portions of the cited reference fail to disclose the subject matter of claim 1. As an example, Appellant submits the cited portions of the cited reference fail to disclose "determining local interface and next neighbor information for the network device." F1 The Examiner cites "(figure 7, items 114, 116, 120, 126, 122, 118, and 124)" of the Hansen reference as allegedly disclosing such feature. F2 However, Appellant notes Figure 7 of the Hansen reference merely purports to be an illustration of a configuration manager GUI, but does not appear to disclose any method steps. F3

As another example, Appellant submits the cited portions of the cited reference fail to disclose "determining whether the local interface and next neighbor information is associated with a logical configuration link stored among a plurality of logical configuration links in a logical link database." F4 While the Examiner cites "(column 5, lines 35-64; Subsystem is a logical link database)," F5

Appellant notes the cited portion of the cited reference states, "The data and programming instruction are stored in the memory subsystem 6...." F6 Appellant sees no teaching as to "determining whether the local interface and next neighbor information is associated with a logical configuration link stored among a plurality of logical configuration links in a logical link database." F7

As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "creating a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database." F8 While the Examiner cites "(column 15, lines 30-50; Unconnected PCI slot are unassociated connection), F9 Appellant notes col. 15, lines 33-39, state, "As may now be seen, the various network entities, as well as unconnected connection interfaces, are graphically displayed on the backplane bitmap 220 using information contained in the bitmap section 36 of the configuration script 12-N and the local configuration file 20 for the Compaq router 122." F10 However, Appellant notes col. 5, lines 49-52, as the Examiner cited in alleging "Subsystem is a logical link database," F11 states "If a particular network device does not have a configuration script, a configuration file cannot be constructed by the network device configuration tool 10." F12 Accordingly, Appellant submits the Examiner's apparent interpretation of the teachings of the prior art would appear to render them inoperable. F13 Thus, Appellant submits the cited portions of the cited reference cannot disclose the subject matter recited in claim 1.

As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "storing the new logical configuration link in the logical link database." F14 While the Examiner cites "(column 13, lines 10-30)," F15 Appellant submits col. 13, lines 22-26, state, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step 156 that the devices/entities cannot be connected." F16 Appellant submits the cited portion of the cited reference fails to disclose "storing the new logical configuration link in the logical link database." F17

As a further example, Appellant submits the cited portions of the cited reference fail to disclose "sending the new logical configuration link to the network device." F18 While the Examiner cites "(column 14, lines 41-60)," F19 Appellant notes col. 14, lines 48-50, states "...before saving the constructed local configuration file 20 to the memory subsystem and associating it with the device." F20 Appellant submits the cited portion of the cited reference does not appear to disclose "sending the new logical configuration link to the network device." F21

Regarding claim 2, Appellant submits the cited portions of the cited reference fail to disclose "selecting a link type." F22 While the Examiner cites "(column 13, lines 1-10; x.25, frame relay, PPP

and HDLC are link types," F23 Appellant submits the Examiner has alleged, with respect to claim 1, from which claim 2 depends, that "Subsystem is a logical link database." F24 Appellant submits the Examiner doesn't provide any evidence that "Subsystem" includes any information pertaining to "frame relay, PPP and HDLC." F25 Thus, Appellant submits the Examiner's apparent interpretation of the teachings of the prior art would appear to render them inoperable. F26

As another example, Appellant submits the cited portions of the cited reference fail to disclose "selecting a link numbering type for the new logical configuration link." F27 While the Examiner alleges "(column 11, lines 13-30; PCI slots are numbered configuration links)," F28 Appellant submits such allegation does not disclose a step of "selecting a link numbering type...." F29

As a further example, Appellant submits the cited portions of the cited reference fail to disclose "selecting a link application for the new logical configuration link." F30 While the Examiner alleges "(column 14, lines 5-25; The script commands are applications; column 13, lines 65-column 14, lines 5)," F31 Appellant notes the Examiner alleged with respect to "creating a new logical configuration link..." of claim 1, from which claim 2 depends, "Unconnected PCI slot are unassociated connection." F32 While the Examiner alleges "The script commands are applications," F31 Appellant sees no allegation by the Examiner that "the script commands" disclose link applications for "unconnected PCI slot," F33 which the Examiner appears to allege disclose "the new logical configuration link." F34 Thus, Appellant submits the Examiner's allegations appear to be inconsistent and would render the purported teachings of the cited reference inoperable. F35

As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "selecting a sub layer interface type for the new logical configuration link." F36 While the Examiner cites "(column 14, lines 15-25; Connection identifiers are configuration links)," F37 Appellant notes the Examiner alleged, with respect to "creating a new logical configuration link" of claim 1, from which claim 2 depends, "Unconnected PCI slot are unassociated connection." F38 Thus, Appellant submits "connection identifiers are configuration links" is inconsistent with the purported teachings alleged by the Examiner with respect to claim 1, thereby apparently rendering such teachings inoperable. F39 Moreover, Appellant submits "connection identifiers are configuration links" fails to disclose "selecting a sub layer interface type...." F40

As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "creating a first endpoint for the new logical configuration link" and "creating a second endpoint for the new logical configuration link." F41 While the Examiner cites "(column 13, lines 10-30)," F42 Appellant submits col. 13, lines 22-26, states, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a

determination is made at step 156 that the devices/entities cannot be connected." F43 Appellant submits the cited portion of the cited reference fails to disclose "creating a first endpoint for the new logical configuration link" F44and "creating a second endpoint for the new logical configuration link." F45

Regarding claim 3, Appellant submits the cited portion of the cited reference fails to disclose "selecting the link type from among a group consisting of: point-to-point, point-to-IP, and point-to-subnet." F46 While the Examiner cites "(column 13, lines 1-10; x.25, frame relay, PPP and HDLC)," F47 Appellant notes the inconsistency Appellant alleges with respect to the Examiner's allegations regarding "selecting a link type" in claim 2, from which claim 3 depends. F48 Thus, Appellant submits the Examiner's allegations with respect to claim 3 also render the Examiner's apparent interpretation of the purported teachings of the cited portions of the cited reference inoperable. F49

Regarding claim 4, Appellant submits the cited portions of the cited reference fail to disclose "selecting the link numbering type from a group consisting of: a numbered type and an unnumbered type." F50 While the Examiner cites "column 11, lines 13-30; PCI slots are numbering type, column 13, lines 28-45; a list of connection interface is un-number type)," F51 Appellant notes the inconsistency Appellant alleges with respect to the Examiner's allegations regarding "selecting a link numbering type..." in claim 2, from which claim 4 depends. F52 Thus, Appellant submits the Examiner's allegations with respect to claim 4 also render the Examiner's apparent interpretation of the purported teachings of the cited portions of the cited reference inoperable. F53

Regarding claim 7, Appellant submits the cited portions of the cited reference fail to disclose "modifying a logical configuration link in the logical link database." F54 While the Examiner cites "(column 11, lines 41-53; Editing is modifying)," F55 Appellant notes the Examiner alleged "Unconnected PCI are unassociated connection" as purportedly teaching "creating a new logical configuration link..." in claim 1, from which claim 7 depends. F56 Appellant sees no reference to such "unconnected PCI" in "(column 11, lines 41-53; Editing is modifying)," as alleged by the Examiner. F57 Thus, Appellant submits the cited portions of the cited reference fail to disclose the subject matter of claim 7.

Regarding claim 8, Appellant submits the cited portions of the cited reference fail to disclose "deleting a logical configuration link in the logical link database." F58 While the Examiner cites "(column 10, lines 1-20)," F59 Appellant notes the Examiner alleged "Unconnected PCI are unassociated connection" as purportedly teaching "creating a new logical configuration link..." in claim 1, from which claim 8 depends. F60 Appellant sees no reference to such "unconnected PCI" in "(column 10, lines 1-20)," as alleged by the Examiner. F61 Moreover, Appellant submits teachings in

"(column 10, lines 1-20)" appear to be inconsistent with "unconnected PCI." F62 For example, "telnet to this device," "view ip addresses," and "view ipx addresses" appear to be inconsistent with "unconnected PCI," as cited by the Examiner with respect to claim 1, from which claim 8 depends. F63 Thus, Appellant submits the cited portions of the cited reference fail to disclose the subject matter of claim 8.

Regarding claim 9, Appellant notes the Examiner states "As per claim 9, it is of the same scope as claim 1. Supra." F64 Appellant respectfully disagrees and notes claim 9 is directed to different subject matter than claim 1. F65 However, to the extent the Examiner relies on the Examiner's rejection of claim 1 to also reject claim 9, Appellant reiterates what Appellant alleges to be the deficiencies of the Examiner's rejection of claim 1, as Appellant discussed above.

Regarding claim 11, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system determines local interface and next neighbor information for the network device." F66 While the Examiner cites "(figure 7, items 114, 116, 120, 126, 122, 118, and 124)," F67 Appellant submits Figure 7 of the Hansen reference merely purports to be an illustration of a configuration manager GUI, but does not appear to disclose "wherein the processing system determines local interface and next neighbor information for the network device." F68

Regarding claim 12, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system determines whether the local interface and next neighbor information is associated with one of the logical configuration links stored in the logical link database." F69 While the Examiner cites "(column 15, lines 30-50; Unconnected PCI slot are unassociated connection)," F70 Appellant submits the "Unconnected PCI slot are unassociated connection" alleged by the Examiner fails to disclose, for example, "next neighbor information" and "the logical link database." F71 Thus, Appellant submits the Examiner has not made a *prima facie* showing of anticipation with respect to the subject matter of claim 12.

Regarding claim 13, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system creates a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links stored in the logical link database." F72 While the Examiner cites "(column 13, lines 10-30)," F73 Appellant submits col. 13, lines 22-26, state, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step 156 that the devices/entities cannot be connected." F74 Appellant submits the cited portion of the cited reference fails to disclose "wherein the processing system creates a new logical configuration link

when the local interface and next neighbor information is not associated with any of the logical configuration links stored in the logical link database." F75

Regarding claim 14, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system causes the new logical configuration link to be stored in the logical link database." F76 While the Examiner cites "(column 13, lines 10-30)," F77 Appellant submits col. 13, lines 22-26, state, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step 156 that the devices/entities cannot be connected." F78 Appellant submits the cited portion of the cited reference fails to disclose "wherein the processing system causes the new logical configuration link to be stored in the logical link database." F79

Regarding claim 16, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system causes the new logical configuration link to be sent to the network device." F80 While the Examiner cites "(column 14, lines 41-60)," F81 Appellant notes col. 14, lines 48-50, states "...before saving the constructed local configuration file 20 to the memory subsystem and associating it with the device." F82 Appellant submits the cited portion of the cited reference does not appear to disclose "wherein the processing system causes the new logical configuration link to be sent to the network device." F83

Regarding claim 17, Appellant submits the cited portions of the cited reference fail to disclose "wherein creating the new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database occurs based on interfaces information entered by a user." F84 The Examiner states "As per claim 17, it is rejected under the same rationale as claim 1. Supra." F85 Appellant respectfully disagrees and notes claim 17 is directed to different subject matter than claim 1. F86 However, to the extent the Examiner relies on the Examiner's rejection of claim 1 to also reject claim 17, Appellant reiterates what Appellant alleges to be the deficiencies of the Examiner's rejection of claim 1, as Appellant discussed above.

Regarding claim 18, Appellant submits the cited portions of the cited reference fail to disclose the subject matter recited in claim 18. F87 Appellant notes the Examiner states "As per claim 18, it is rejected under the same rationale as claim 2." F88 Appellant respectfully disagrees and notes claim 18 is directed to different subject matter than claim 2. F89 To the extent the Examiner relies on the Examiner's rejection of claim 2 to also reject claim 18, Appellant reiterates what Appellant alleges to be the deficiencies of the Examiner's rejection of claim 2, as Appellant discussed above. Nonetheless, Appellant submits the Examiner has not alleged anticipation with respect to subject matter recited in

claim 18. F90 As one example, Appellant submits claim 18 recites "populating form panels with the link type, the link numbering type, the link application, and the sub layer interface type," F91 while claim 2 does not. F92 As another example, Appellant submits claim 18 recites "receiving user input of interfaces information." F93 As yet another example, Appellant submits claim 18 recites "validating the interfaces information." F94 As a further example, Appellant submits claim 18 recites "creating a link in accordance with the interfaces information." F95 As another example, Appellant submits claim 18 recites "provisioning the link." F96 Appellant submits the Examiner has not alleged any teaching as to such subject matter. F97 Thus, Appellant submits the Examiner has not made a *prima facie* showing of anticipation with respect to claim 18.

Rejection of Claim 5 under 35 U.S.C. 103(a) as being unpatentable over Hansen in view of Hansen US Patent 5,838,907 [sic] in view of Hardwick US Patent 5,550,816

MPEP § 2143 describes examples of basic requirements of a *prima facie* case of obviousness under 35 U.S.C. § 103. As Appellant describes in detail below, Appellant submits there exist clear errors in the Examiner's rejections and/or the Examiner's omissions of one or more aspects of a *prima facie* rejection.

Regarding claim 5, Appellant submits the cited portions of the cited reference fail to render unpatentable the subject matter of claim 5. As an example, Appellant submits the cited portions of the cited reference fail to disclose or suggest "selecting the link application from a group consisting of: Internet Protocol Forwarding, Multi-Protocol Label Switching and Internet Protocol Forwarding, and Multi-Protocol Label Switching." F98 Appellant notes the Examiner alleges, with respect to claim 2, from which claim 5 depends, "The script commands are applications." F99 However, the Examiner now alleges "Hardwick teaches the step of selecting a link application from a group consisting of: Internet Protocol Forwarding, Multi-Protocol Label Switching and Internet Protocol Forwarding, and Multi-Protocol Label Switching (column 43, lines 60-column 44, lines 5)." F100 Appellant submits the Examiner's allegations as to purported teachings of the cited references with respect to the subject matter of claims 2 and 5 are inconsistent and contradictory, thereby rendering the supposed combination of the purported teachings inoperable. F101 Moreover, Appellant submits the Examiner's alleged motivation to combine the references does not appear to pertain to the supposed combination of the purported teachings. F102 Thus, Appellant submits the Examiner has not made a *prima facie* showing of obviousness with respect to the subject matter of claim 5.

Rejection of claim 6 under 35 U.S.C. 103(a) as being unpatentable over Hansen in view of Hansen US Patent 5,838,907 [sic] in view of Chui US Patent 2002/0165978

MPEP § 2143 describes examples of basic requirements of a *prima facie* case of obviousness under 35 U.S.C. § 103. Appellant submits there exist clear errors in the Examiner's rejections and/or the Examiner's omissions of one or more aspects of a *prima facie* rejection.

Regarding claim 6, Appellant submits the cited portions of the cited reference fail to render unpatentable the subject matter of claim 6. As an example, Appellant submits the cited portions of the cited reference fail to disclose or suggest "selecting the sub layer interface type from a group consisting of: Packet Over Sonet, Asynchronous Transfer Mode, and GigEthernet." F103 Appellant notes the Examiner alleges, with respect to claim 2, from which claim 6 depends, "Connection identifiers are configuration links." F104 However, the Examiner now alleges "Chui teaches selecting a sub layer interface type comprises the step of: Selecting the sub-layer interface type from a group consisting of: Packet over Sonet, Asynchronous Transfer Mode, and GigEthernet." F105 Appellant submits the Examiner's allegations as to purported teachings of the cited references with respect to the subject matter of claims 2 and 6 are inconsistent and contradictory, thereby rendering the supposed combination of the purported teachings inoperable. F106 Moreover, Appellant submits the Examiner's alleged motivation to combine the references does not appear to pertain to the supposed combination of the purported teachings. F107 Thus, Appellant submits the Examiner has not made a *prima facie* showing of obviousness with respect to the subject matter of claim 6.

Facts referenced above:

F1. As an example, Appellant submits the cited portions of the cited reference fail to disclose "determining local interface and next neighbor information for the network device." Office action response 11/20/2008, p. 6, l. 7-9.

F2. The Examiner cites "(figure 7, items 114, 116, 120, 126, 122, 118, and 124)" of the Hansen reference as allegedly disclosing such feature. Office action response 11/20/2008, p. 6, l. 9-10.

F3. Appellant notes Figure 7 of the Hansen reference merely purports to be an illustration of a configuration manager GUI, but does not appear to disclose any method steps. Office action response 11/20/2008, p. 6, l. 10-12; Hansen, Fig. 7.

F4. As another example, Appellant submits the cited portions of the cited reference fail to disclose "determining whether the local interface and next neighbor information is associated with a logical configuration link stored among a plurality of logical configuration links in a logical link database." Office action response 11/20/2008, p. 6, l. 13-15.

F5. The Examiner cites "(column 5, lines 35-64; Subsystem is a logical link database)." Office action response 11/20/2008, p. 6, l. 16.

F6. Appellant notes the cited portion of the cited reference states, "The data and programming instruction are stored in the memory subsystem 6…." Office action response 11/20/2008, p. 6, l. 16-18; Hansen, col. 5, lines 35-64.

F7. Appellant sees no teaching as to "determining whether the local interface and next neighbor information is associated with a logical configuration link stored among a plurality of logical configuration links in a logical link database." Office action response 11/20/2008, p. 6, l. 18-20.

F8. As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "creating a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database." Office action response 11/20/2008, p. 6, l. 21-23.

F9. The Examiner cites "(column 15, lines 30-50; Unconnected PCI slot are unassociated connection). Office action response 11/20/2008, p. 6, l. 24-25.

F10. Appellant notes col. 15, lines 33-39, state, "As may now be seen, the various network entities, as well as unconnected connection interfaces, are graphically displayed on the backplane bitmap 220 using information contained in the bitmap section 36 of the configuration script 12-N and the local configuration file 20 for the Compaq router 122." Office action response 11/20/2008, p. 6, l. 25-28; Hansen, col. 15, lines 33-39.

F11. The Examiner cited col. 5, lines 49-52, in alleging "Subsystem is a logical link database." Office action response 11/20/2008, p. 6, l. 28-29.

F12. Appellant submits col. 5, lines 49-52, states "If a particular network device does not have a configuration script, a configuration file cannot be constructed by the network device configuration tool 10." Office action response 11/20/2008, p. 6, l. 29, to p. 7, l. 2; Hansen, col. 5, lines 49-52.

F13. Appellant submits the Examiner's apparent interpretation of the teachings of the prior art would appear to render them inoperable. Office action response 11/20/2008, p. 7, l. 2-3.

F14. As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "storing the new logical configuration link in the logical link database." Office action response 11/20/2008, p. 7, l. 5-6.

F15. The Examiner cites "(column 13, lines 10-30)." Office action response 11/20/2008, p. 7, l. 6-7.

F16. Appellant submits col. 13, lines 22-26, state, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step 156 that the devices/entities cannot be connected." Office action response 11/20/2008, p. 7, l. 7-9; Hansen, col. 13, lines 22-26.

F17. Appellant submits the cited portion of the cited reference fails to disclose "storing the new logical configuration link in the logical link database." Office action response 11/20/2008, p. 7, l. 9-11.

F18. As a further example, Appellant submits the cited portions of the cited reference fail to disclose "sending the new logical configuration link to the network device." Office action response 11/20/2008, p. 7, l. 12-13.

F19. The Examiner cites "(column 14, lines 41-60)." Office action response 11/20/2008, p. 7, l. 13-14.

F20. Appellant notes col. 14, lines 48-50, states "...before saving the constructed local configuration file 20 to the memory subsystem and associating it with the device." Office action response 11/20/2008, p. 7, l. 14-15; Hansen, col. 14, lines 48-50.

F21. Appellant submits the cited portion of the cited reference does not appear to disclose "sending the new logical configuration link to the network device." Office action response 11/20/2008, p. 7, l. 16-17.

F22. Regarding claim 2, Appellant submits the cited portions of the cited reference fail to disclose "selecting a link type." Office action response 11/20/2008, p. 7, l. 19-20.

F23. The Examiner cites "(column 13, lines 1-10; x.25, frame relay, PPP and HDLC are link types." Office action response 11/20/2008, p. 7, l. 20-21.

F24. Appellant submits the Examiner has alleged, with respect to claim 1, from which claim 2 depends, that "Subsystem is a logical link database." Office action response 11/20/2008, p. 7, l. 21-22; final Office action 3/18/2009, p. 2, l. 22.

F25. Appellant submits the Examiner doesn't provide any evidence that "Subsystem" includes any information pertaining to "frame relay, PPP and HDLC." Office action response 11/20/2008, p. 7, l. 22-24.

F26. Appellant submits the Examiner's apparent interpretation of the teachings of the prior art would appear to render them inoperable. Office action response 11/20/2008, p. 7, l. 24-25.

F27. Appellant submits the cited portions of the cited reference fail to disclose "selecting a link numbering type for the new logical configuration link." Office action response 11/20/2008, p. 7, l. 26-27.

F28. The Examiner alleges "(column 11, lines 13-30; PCI slots are numbered configuration links)." Office action response 11/20/2008, p. 7, l. 27-28.

F29. Appellant submits such allegation does not disclose a step of "selecting a link numbering type...." Office action response 11/20/2008, p. 7, l. 28-29.

F30. As a further example, Appellant submits the cited portions of the cited reference fail to disclose "selecting a link application for the new logical configuration link." Office action response 11/20/2008, p. 8, l. 1-2.

F31. The Examiner alleges "(column 14, lines 5-25; The script commands are applications; column 13, lines 65-column 14, lines 5)." Office action response 11/20/2008, p. 8, l. 2-4; final Office action 3/18/2009, p. 3, l. 18.

F32. Appellant notes the Examiner alleged with respect to "creating a new logical configuration link..." of claim 1, from which claim 2 depends, "Unconnected PCI slot are unassociated connection." Office action response 11/20/2008, p. 8, l. 4-5; final Office action 3/18/2009, p. 3, l. 3.

F33. Appellant sees no allegation by the Examiner that "the script commands" disclose link applications for "unconnected PCI slot." Office action response 11/20/2008, p. 8, l. 6-7.

F34. The Examiner appears to allege "unconnected PCI slot" discloses "the new logical configuration link." Office action response 11/20/2008, p. 8, l. 7-8; final Office action 3/18/2009 p. 3, l. 1-3.

F35. Appellant submits the Examiner's allegations appear to be inconsistent and would render the purported teachings of the cited reference inoperable. Office action response 11/20/2008, p. 8, l. 8-10.

F36. As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "selecting a sub layer interface type for the new logical configuration link." Office action response 11/20/2008, p. 8, l. 11-12.

F37. The Examiner cites "(column 14, lines 15-25; Connection identifiers are configuration links)." Office action response 11/20/2008, p. 8, l. 13; final Office action 3/18/2009, p. 3, l. 19-20.

F38. Appellant notes the Examiner alleged, with respect to "creating a new logical configuration link" of claim 1, from which claim 2 depends, "Unconnected PCI slot are unassociated connection." Office action response 11/20/2008, p. 8, l. 13-15; final Office action 3/18/2009, p. 3, l. 1-3.

F39. Appellant submits "connection identifiers are configuration links" is inconsistent with the purported teachings alleged by the Examiner with respect to claim 1, thereby apparently rendering such teachings inoperable. Office action response 11/20/2008, p. 8, l. 15-17.

F40. Appellant submits "connection identifiers are configuration links" fails to disclose "selecting a sub layer interface type...." Office action response 11/20/2008, p. 8, l. 18-19.

F41. As yet another example, Appellant submits the cited portions of the cited reference fail to disclose "creating a first endpoint for the new logical configuration link" and "creating a second endpoint for the new logical configuration link." Office action response 11/20/2008, p. 8, l. 20-22.

F42. The Examiner cites "(column 13, lines 10-30)." Office action response 11/20/2008, p. 8, l. 22; final Office action 3/18/2009, p. 4, l. 1-2.

F43. Appellant submits col. 13, lines 22-26, states, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step 156 that the devices/entities cannot be connected." Office action response 11/20/2008, p. 8, l. 23-25; Hansen, col. 13, lines 22-26.

F44. Appellant submits the cited portion of the cited reference fails to disclose "creating a first endpoint for the new logical configuration link." Office action response 11/20/2008, p. 8, l. 25-26.

F45. Appellant submits the cited portion of the cited reference fails to disclose "creating a second endpoint for the new logical configuration link." Office action response 11/20/2008, p. 8, l. 25-27.

F46. Regarding claim 3, Appellant submits the cited portion of the cited reference fails to disclose "selecting the link type from among a group consisting of: point-to-point, point-to-IP, and point-to-subnet." Office action response 11/20/2008, p. 8, l. 29-30.

F47. The Examiner cites "(column 13, lines 1-10; x.25, frame relay, PPP and HDLC)." Office action response 11/20/2008, p. 8, l. 30, to p. 9, l. 1; final Office action 3/18/2009, p. 4, l. 6.

F48. Appellant notes the inconsistency Appellant alleges with respect to the Examiner's allegations regarding "selecting a link type" in claim 2, from which claim 3 depends. Office action response 11/20/2008, p. 9, l. 1-3.

F49. Appellant submits the Examiner's allegations with respect to claim 3 also render the Examiner's apparent interpretation of the purported teachings of the cited portions of the cited reference inoperable. Office action response 11/20/2008, p. 9, l. 3-5.

F50. Regarding claim 4, Appellant submits the cited portions of the cited reference fail to disclose "selecting the link numbering type from a group consisting of: a numbered type and an unnumbered type." Office action response 11/20/2008, p. 9, l. 7-9.

F51. The Examiner cites "column 11, lines 13-30; PCI slots are numbering type, column 13, lines 28-45; a list of connection interface is un-number type)." Office action response 11/20/2008, p. 9, l. 9-10; final Office action 3/18/2009, p. 4, l. 10-11.

F52. Appellant notes the inconsistency Appellant alleges with respect to the Examiner's allegations regarding "selecting a link numbering type..." in claim 2. Office action response 11/20/2008, p. 9, l. 10-12.

F53. Appellant submits the Examiner's allegations with respect to claim 4 also render the Examiner's apparent interpretation of the purported teachings of the cited portions of the cited reference inoperable. Office action response 11/20/2008, p. 9, l. 12-14.

F54. Regarding claim 7, Appellant submits the cited portions of the cited reference fail to disclose "modifying a logical configuration link in the logical link database." Office action response 11/20/2008, p. 9, l. 16-17.

F55. The Examiner cites "(column 11, lines 41-53; Editing is modifying)." Office action response 11/20/2008, p. 9, l. 17-18; final Office action 3/18/2009, p. 4, l. 13-14.

F56. The Examiner alleged "Unconnected PCI are unassociated connection" as purportedly teaching "creating a new logical configuration link..." in claim 1. Office action response 11/20/2008, p. 9, l. 18-20; final Office action 3/18/2009, p. 3, l. 1-3.

F57. Appellant sees no reference to such "unconnected PCI" in "(column 11, lines 41-53; Editing is modifying)," as alleged by the Examiner. Office action response 11/20/2008, p. 9, l. 21.

F58. Regarding claim 8, Appellant submits the cited portions of the cited reference fail to disclose "deleting a logical configuration link in the logical link database." Office action response 11/20/2008, p. 9, l. 24-25.

F59. The Examiner cites "(column 10, lines 1-20)." Office action response 11/20/2008, p. 9, l. 25-26; final Office action 3/18/2009, p. 4, l. 16.

F60. Appellant notes the Examiner alleged "Unconnected PCI are unassociated connection" as purportedly teaching "creating a new logical configuration link..." in claim 1. Office action response 11/20/2008, p. 9, l. 26-27; final Office action 3/18/2009, p. 3, l. 1-3.

F61. Appellant sees no reference to such "unconnected PCI" in "(column 10, lines 1-20)," as alleged by the Examiner. Office action response 11/20/2008, p. 9, l. 28-29.

F62. Appellant submits teachings in "(column 10, lines 1-20)" appear to be inconsistent with "unconnected PCI." Office action response 11/20/2008, p. 9, l. 29-30.

F63. For example, "telnet to this device," "view ip addresses," and "view ipx addresses" appear to be inconsistent with "unconnected PCI," as cited by the Examiner with respect to claim 1. Office action response 11/20/2008, p. 9, l. 30, to p. 10, l. 2; Hansen, column 10, lines 1-20.

F64. Regarding claim 9, Appellant notes the Examiner states "As per claim 9, it is of the same scope as claim 1. Supra." Office action response 11/20/2008, p. 10, l. 5-6; final Office action 3/18/2009, p. 4, l. 17.

F65. Appellant respectfully disagrees and notes claim 9 is directed to different subject matter than claim 1. Office action response 11/20/2008, p. 10, l. 6-7.

F66. Regarding claim 11, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system determines local interface and next neighbor information for the network device." Office action response 11/20/2008, p. 10, l. 13-15.

F67. The Examiner cites "(figure 7, items 114, 116, 120, 126, 122, 118, and 124)." Office action response 11/20/2008, p. 10, l. 15; final Office action 3/18/2009, p. 5, l. 5.

F68. Appellant submits Figure 7 of the Hansen reference merely purports to be an illustration of a configuration manager GUI, but does not appear to disclose "wherein the processing system determines local interface and next neighbor information for the network device." Office action response 11/20/2008, p. 10, l. 16-18; Hansen, Fig. 7.

F69. Regarding claim 12, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system determines whether the local interface and next neighbor information is associated with one of the logical configuration links stored in the logical link database." Office action response 11/20/2008, p. 10, l. 20-22.

F70. The Examiner cites "(column 15, lines 30-50; Unconnected PCI slot are unassociated connection)." Office action response 11/20/2008, p. 10, l. 22-23; final Office action 3/18/2009, p. 5, l. 8-9.

F71. Appellant submits the "Unconnected PCI slot are unassociated connection" alleged by the Examiner fails to disclose, for example, "next neighbor information" and "the logical link database." Office action response 11/20/2008, p. 10, l. 24-25.

F72. Regarding claim 13, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system creates a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links stored in the logical link database." Office action response 11/20/2008, p. 11, l. 1-4.

F73. The Examiner cites "(column 13, lines 10-30)." Office action response 11/20/2008, p. 11, l. 4; final Office action 3/18/2009, p. 4, l. 13.

F74. Appellant submits col. 13, lines 22-26, state, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step 156 that the devices/entities cannot be connected." Office action response 11/20/2008, p. 11, l. 4-7.

F75. Appellant submits the cited portion of the cited reference fails to disclose "wherein the processing system creates a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links stored in the logical link database." Office action response 11/20/2008, p. 11, l. 7-10.

F76. Regarding claim 14, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system causes the new logical configuration link to be stored in the logical link database." Office action response 11/20/2008, p. 11, l. 12-14.

F77. The Examiner cites "(column 13, lines 10-30)." Office action response 11/20/2008, p. 11, l. 14; final Office action 3/18/2009, p. 5, l. 16.

F78. Appellant submits col. 13, lines 22-26, state, "If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step 156 that the devices/entities cannot be connected." Office action response 11/20/2008, p. 11, l. 14-17; Hansen, col. 13, lines 22-26.

F79. Appellant submits the cited portion of the cited reference fails to disclose "wherein the processing system causes the new logical configuration link to be stored in the logical link database." Office action response 11/20/2008, p. 11, l. 17-19.

F80. Regarding claim 16, Appellant submits the cited portions of the cited reference fail to disclose "wherein the processing system causes the new logical configuration link to be sent to the network device." Office action response 11/20/2008, p. 11, l. 23-25.

F81. The Examiner cites "(column 14, lines 41-60)." Office action response 11/20/2008, p. 11, l. 25; final Office action 3/18/2009, p. 5, l. 21.

F82. Appellant notes col. 14, lines 48-50, states "...before saving the constructed local configuration file 20 to the memory subsystem and associating it with the device." Office action response 11/20/2008, p. 11, l. 25-27; Hansen, col. 14, lines 48-50.

F83. Appellant submits the cited portion of the cited reference does not appear to disclose "wherein the processing system causes the new logical configuration link to be sent to the network device." Office action response 11/20/2008, p. 11, l. 27-29.

F84. Regarding claim 17, Appellant submits the cited portions of the cited reference fail to disclose "wherein creating the new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database occurs based on interfaces information entered by a user." Office action response 11/20/2008, p. 12, l. 1-4.

F85. The Examiner states "As per claim 17, it is rejected under the same rationale as claim 1. Supra." Office action response 11/20/2008, p. 12, l. 4-5; final Office action 3/18/2009, p. 6, l. 1.

F86. Appellant notes claim 17 is directed to different subject matter than claim 1. Office action response 11/20/2008, p. 12, l. 5-6.

F87. Regarding claim 18, Appellant submits the cited portions of the cited reference fail to disclose the subject matter recited in claim 18. Office action response 11/20/2008, p. 12, l. 10-11.

F88. Appellant notes the Examiner states "As per claim 18, it is rejected under the same rationale as claim 2." Office action response 11/20/2008, p. 12, l. 11-12; final Office action 3/18/2009, p. 6, l. 2.

F89. Appellant notes claim 18 is directed to different subject matter than claim 2. Office action response 11/20/2008, p. 12, l. 12-13.

F90. Nonetheless, Appellant submits the Examiner has not alleged anticipation with respect to subject matter recited in claim 18. Office action response 11/20/2008, p. 12, l. 16-17.

F91. As one example, Appellant submits claim 18 recites "populating form panels with the link type, the link numbering type, the link application, and the sub layer interface type." Office action response 11/20/2008, p. 12, l. 17-18.

F92. Claim 2 does not. Office action response 11/20/2008, p. 12, 18-19.

F93. As another example, Appellant submits claim 18 recites "receiving user input of interfaces information." Office action response 11/20/2008, p. 12, l. 19-20.

F94. As yet another example, Appellant submits claim 18 recites "validating the interfaces information." Office action response 11/20/2008, p. 12, l. 20-21.

F95. As a further example, Appellant submits claim 18 recites "creating a link in accordance with the interfaces information." Office action response 11/20/2008, p. 12, l. 21-22.

F96. As another example, Appellant submits claim 18 recites "provisioning the link." Office action response 11/20/2008, p. 12, l. 22-23.

F97. Appellant submits the Examiner has not alleged any teaching as to such subject matter. Office action response 11/20/2008, p. 12, l. 23-24.

F98. As an example, Appellant submits the cited portions of the cited reference fail to disclose or suggest "selecting the link application from a group consisting of: Internet Protocol Forwarding, Multi-Protocol Label Switching and Internet Protocol Forwarding, and Multi-Protocol Label Switching." Office action response 11/20/2008, p. 13, l. 2-5.

F99. Appellant notes the Examiner alleges, with respect to claim 2, "The script commands are applications." Office action response 11/20/2008, p. 13, l. 5-6; final Office action 3/18/2009, p. 3, l. 18.

F100. The Examiner alleges "Hardwick teaches the step of selecting a link application from a group consisting of: Internet Protocol Forwarding, Multi-Protocol Label Switching and Internet Protocol Forwarding, and Multi-Protocol Label Switching (column 43, lines 60-column 44, lines 5)." Office action response 11/20/2008, p. 13, l. 6-9; final Office action 3/18/2009, p. 6, l. 17-19.

F101. Appellant submits the Examiner's allegations as to purported teachings of the cited references with respect to the subject matter of claims 2 and 5 are inconsistent and contradictory, thereby rendering the supposed combination of the purported teachings inoperable. Office action response 11/20/2008, p. 13, l. 9-12.

F102. Appellant submits the Examiner's alleged motivation to combine the references does not appear to pertain to the supposed combination of the purported teachings. Office action response 11/20/2008, p. 13, l. 12-14.

F103. As an example, Appellant submits the cited portions of the cited reference fail to disclose or suggest "selecting the sub layer interface type from a group consisting of: Packet Over Sonet, Asynchronous Transfer Mode, and GigEthernet." Office action response 11/20/2008, p. 13, l. 21-23.

F104. Appellant notes the Examiner alleges, with respect to claim 2, from which claim 6 depends, "Connection identifiers are configuration links." Office action response 11/20/2008, p. 13, l. 23-25; final Office action 3/18/2009, p. 3, l. 20.

F105. The Examiner alleges "Chui teaches selecting a sub layer interface type comprises the step of: Selecting the sub-layer interface type from a group consisting of: Packet over Sonet, Asynchronous Transfer Mode, and GigEthernet." Office action response 11/20/2008, p. 13, l. 25-27; final Office action, p. 7, l. 7-9.

F106. Appellant submits the Examiner's allegations as to purported teachings of the cited references with respect to the subject matter of claims 2 and 6 are inconsistent and contradictory, thereby rendering the supposed combination of the purported teachings inoperable. Office action response 11/20/2008, p. 13, l. 27-30.

F107. Appellant submits the Examiner's alleged motivation to combine the references does not appear to pertain to the supposed combination of the purported teachings. Office action response 11/20/2008, p. 13, l. 30, to p. 14, l. 2.

CLAIMS APPENDIX

1. A network administration method for provisioning logical configuration links for at least two network devices through a dedicated graphical user interface form, the method comprising:

selecting a network device having at least one network interface through the dedicated graphical user interface form;

determining local interface and next neighbor information for the network device;

determining whether the local interface and next neighbor information is associated with a logical configuration link stored among a plurality of logical configuration links in a logical link database;

creating a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database;

storing the new logical configuration link in the logical link database;

validating the new logical configuration link;

sending the new logical configuration link to the network device; and

displaying a graphical representation of the new logical configuration link on a display device.

2. The method of claim 1, wherein the step of creating a new logical configuration link further comprises the steps of:

selecting a link type;

selecting a link numbering type for the new logical configuration link;

selecting a link application for the new logical configuration link;

selecting a sub layer interface type for the new logical configuration link;

creating a first endpoint for the new logical configuration link; and

creating a second endpoint for the new logical configuration link.

3. The method of claim 2, wherein the step of selecting the link type further comprises the step of:

selecting the link type from among a group consisting of: point-to-point, point-to-IP, and point-to-subnet.

4. The method of claim 2, wherein the step of selecting a link numbering type further comprises the step of:

selecting the link numbering type from a group consisting of: a numbered type and an unnumbered type.

5. The method of claim 2, wherein the step of selecting a link application further comprises the step of:

selecting the link application from a group consisting of: Internet Protocol Forwarding, Multi-Protocol Label Switching and Internet Protocol Forwarding, and Multi-Protocol Label Switching.

6. The method of claim 2, wherein the step of selecting a sub layer interface type further comprises the step of:

selecting the sub layer interface type from a group consisting of: Packet Over Sonet, Asynchronous Transfer Mode, and GigEthernet.

7. The method of claim 1, further comprising the step of:

modifying a logical configuration link in the logical link database.

8. The method of claim 1, further comprising the step of:

deleting a logical configuration link in the logical link database.

9. Apparatus for provisioning logical configuration links comprising:

a logical link database for storing logical configuration links;

a processing system coupled to the logical link database for accessing the logical link database; and

a display device coupled to the processing system for displaying a graphical user interface form comprising a graphical representation of a logical configuration link.

10. The apparatus of claim 9 wherein the display device provides an ability to select a network device having at least one network interface through the graphical user interface form.

11. The apparatus of claim 9 wherein the processing system determines local interface and next neighbor information for the network device.

12. The apparatus of claim 11 wherein the processing system determines whether the local

interface and next neighbor information is associated with one of the logical configuration links stored in the logical link database.

13. The apparatus of claim 12 wherein the processing system creates a new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links stored in the logical link database.

14. The apparatus of claim 13 wherein the processing system causes the new logical configuration link to be stored in the logical link database.

15. The apparatus of claim 14 wherein the processing system validates the new logical configuration link.

16. The apparatus of claim 15 wherein the processing system causes the new logical configuration link to be sent to the network device.

17. The method of claim 1 wherein creating the new logical configuration link when the local interface and next neighbor information is not associated with any of the logical configuration links in the logical link database occurs based on interfaces information entered by a user.

18. A method comprising:

selecting a link type;

selecting a link numbering type;

selecting a link application;

selecting a sub layer interface type;

creating a first endpoint;

creating a second endpoint;

populating form panels with the link type, the link numbering type, the link application, and the sub layer interface type;

receiving user input of interfaces information;

validating the interfaces information;

creating a link in accordance with the interfaces information; and

provisioning the link.

EVIDENCE APPENDIX

As presently advised, no evidence was submitted pursuant to 37 C.F.R. §§ 1.130, 1.131, or 1.132. In the Notice of References Cited (Form PTO-892) included with the Office action mailed 03-18-2009, the Examiner cited U.S. Patent No. 5,838,907, issued to Hansen, U.S. Patent No. 5,550,816, issued to Hardwick and U.S. Patent Publication No. 2002/0165978 to Chui. As the Examiner relied upon the following evidence in support of final rejection, Appellant relies upon such evidence in the appeal: U.S. Patent No. 5,838,907, issued to Hansen, U.S. Patent No. 5,550,816, issued to Hardwick and U.S. Patent Publication No. 2002/0165978 to Chui, copies of which are provided below.

RELATED PROCEEDINGS APPENDIX

As stated above, as presently advised, there are no other prior or pending appeals, interferences, or judicial proceedings known to Appellant, the Appellant's legal representative, or Assignee which may be related to, directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal. Thus, no copies of decisions rendered by a court or by the Board are provided.

Respectfully submitted,

06/11/2010 Date

Ross D. Snyder, Reg. No. 37,730
Attorney for Appellant(s)
Ross D. Snyder & Associates, Inc.
115 Wild Basin Road, Suite 107
Austin, Texas 78746
(512) 347-9223 (phone)
(512) 347-9224 (fax)

US005838907A

United States Patent [19]

Hansen

[11] **Patent Number: 5,838,907**

[45] **Date of Patent: Nov. 17, 1998**

[54] **CONFIGURATION MANAGER FOR NETWORK DEVICES AND AN ASSOCIATED METHOD FOR PROVIDING CONFIGURATION INFORMATION THERETO**

[75] Inventor: **Peter A. Hansen**, Houston, Tex.

[73] Assignee: **Compaq Computer Corporation**, Houston, Tex.

[21] Appl. No.: **603,062**

[22] Filed: **Feb. 20, 1996**

[51] **Int. Cl.**[6] **G06F 9/00**

[52] **U.S. Cl.** **395/200.5**; 395/200.51; 395/200.53; 395/200.54; 395/200.55

[58] **Field of Search** 395/200.5, 200.51, 395/500.47, 200.81, 200.82, 200.53, 200.54, 200.55

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,864,492	9/1989	Blakey-Fogel et al.	706/45
5,257,387	10/1993	Richek et al.	395/800
5,261,044	11/1993	Dev et al.	345/537
5,353,401	10/1994	Iizawa et al.	395/161
5,353,432	10/1994	Richek et al.	395/500
5,394,522	2/1995	Sanchez-Frank et al.	395/159
5,438,528	8/1995	Emerson et al.	364/580
5,452,415	9/1995	Hotka	395/161
5,491,796	2/1996	Wanderer et al.	200/54
5,500,934	3/1996	Austin et al.	345/326

FOREIGN PATENT DOCUMENTS

490 624 A2	6/1992	European Pat. Off.	G06F 15/16
2 278 468 A	11/1994	European Pat. Off.	G06F 9/445
2 206 713	1/1989	United Kingdom	G06F 15/60
WO 94/10645	5/1994	WIPO	G06F 15/62

OTHER PUBLICATIONS

"HP Router manager—Getting Start Guide", Hewlett Packard, Mar. 1995.

"Architecture for Graphic Network Install Interface," IBM Technical Disclosure Bulletin, vol. 38, No. 10, Oct. 1995, pp. 465–467, Armonk, NY, USA.

CiscoWorks for Windows Features, Sep. 1994, pp. 1–20.

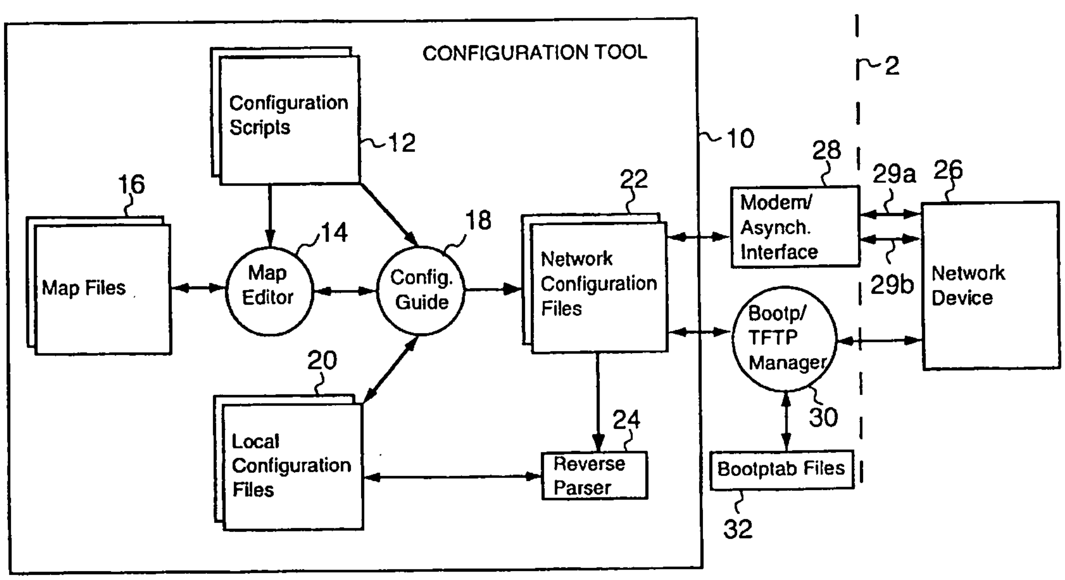
Managing Cisco Device Configurations, CiscoWorks User Guide, Dec. 1994, pp. 1–1 to 1–84.

Primary Examiner—Alyssa H. Bowler
Assistant Examiner—Dzung Nguyen
Attorney, Agent, or Firm—Beyer & Weaver, LLP

[57] **ABSTRACT**

A configuration manager for configuring a network device remotely coupled thereto and an associated computer-implemented method for configuring the network device. The configuration manager includes a configuration script stored in a memory subsystem of a computer system and first and second software modules respectively executable by a processor subsystem of the computer system. The configuration script contains a series of executable instructions for constructing a configuration file and a bootptab file for a first specified type of network device. By executing the instructions contained in the configuration script, the first software module may construct a configuration file suitable for upload to a network device and a bootptab file suitable for identifying the network device. Configuration requests issued by the network device are processed by the second software module by identifying the requesting network device using the constructed bootptab file and configuring the requesting network device by uploading the constructed configuration file thereto.

18 Claims, 12 Drawing Sheets



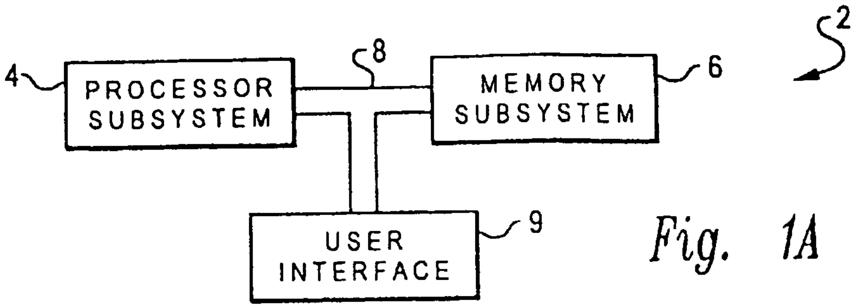


Fig. 1A

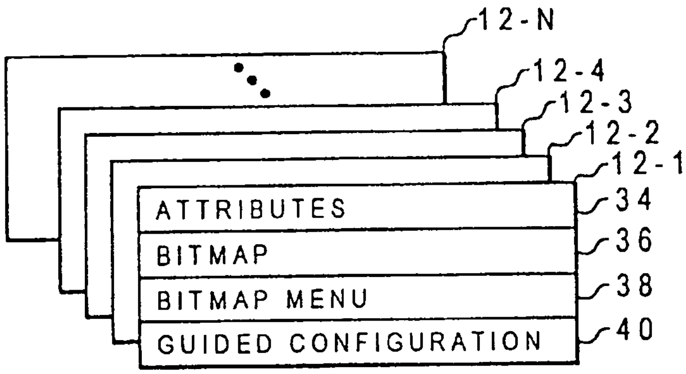


Fig. 2A

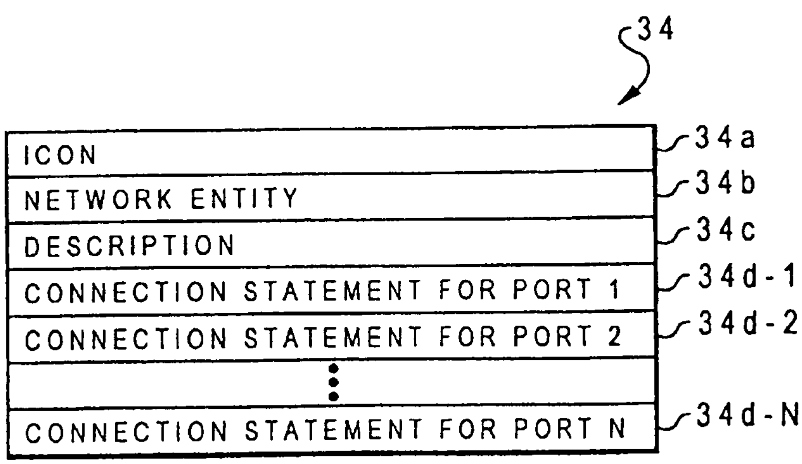


Fig. 2B

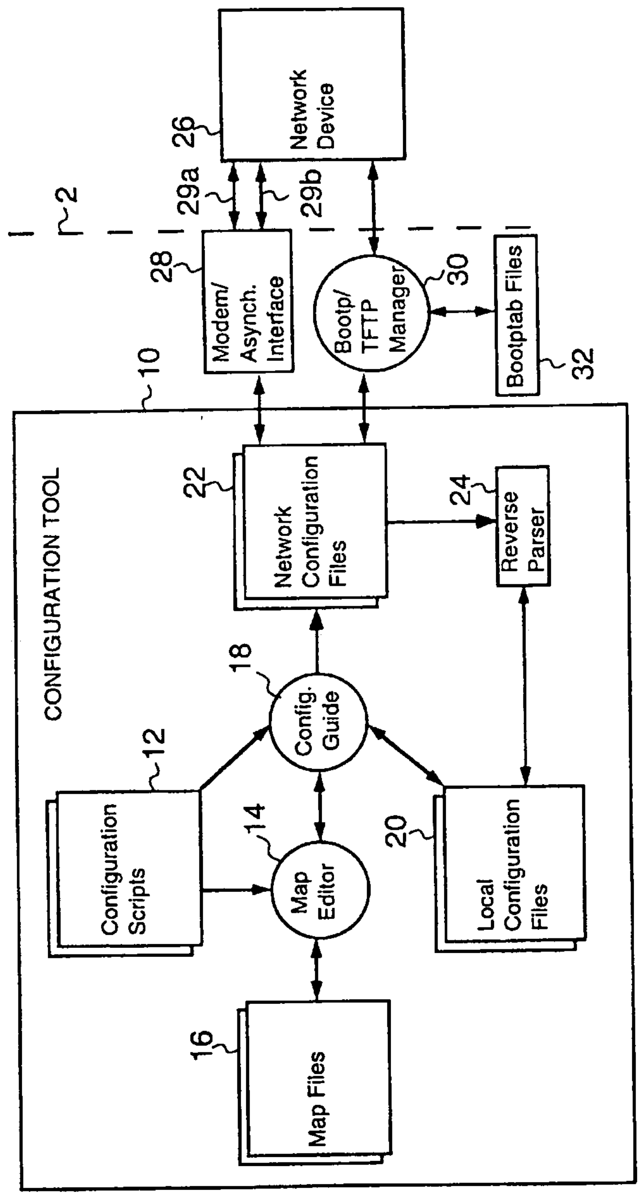

CONFIGURATION TOOL
10
Configuration Scripts
12
Map Editor
14
Map Files
16
Config. Guide
18
Local Configuration Files
20
Network Configuration Files
22
Reverse Parser
24
Network Device
26
Modem/ Asynch. Interface
28
29a
29b
2
Bootp/ TFTP Manager
30
Bootptab Files
32

Fig. 1B

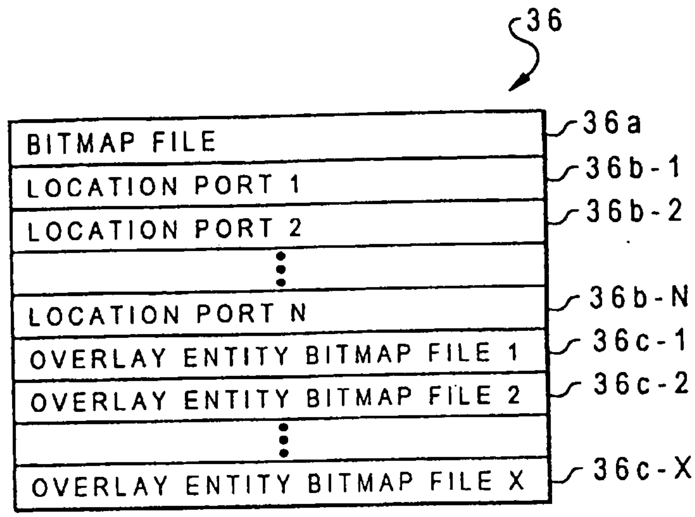


Fig. 2C

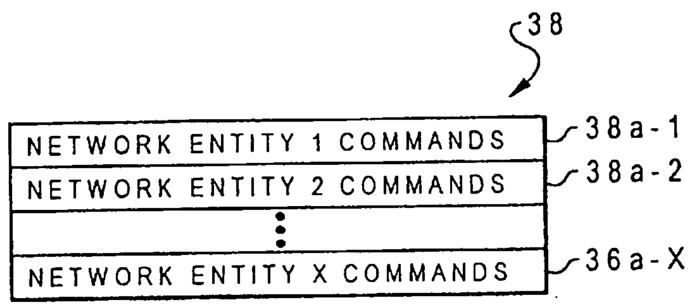


Fig. 2D

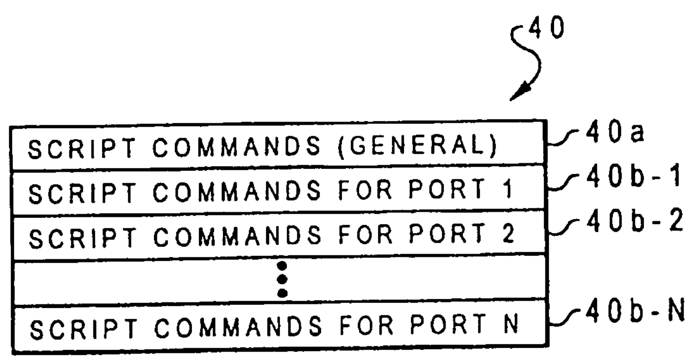


Fig. 2E

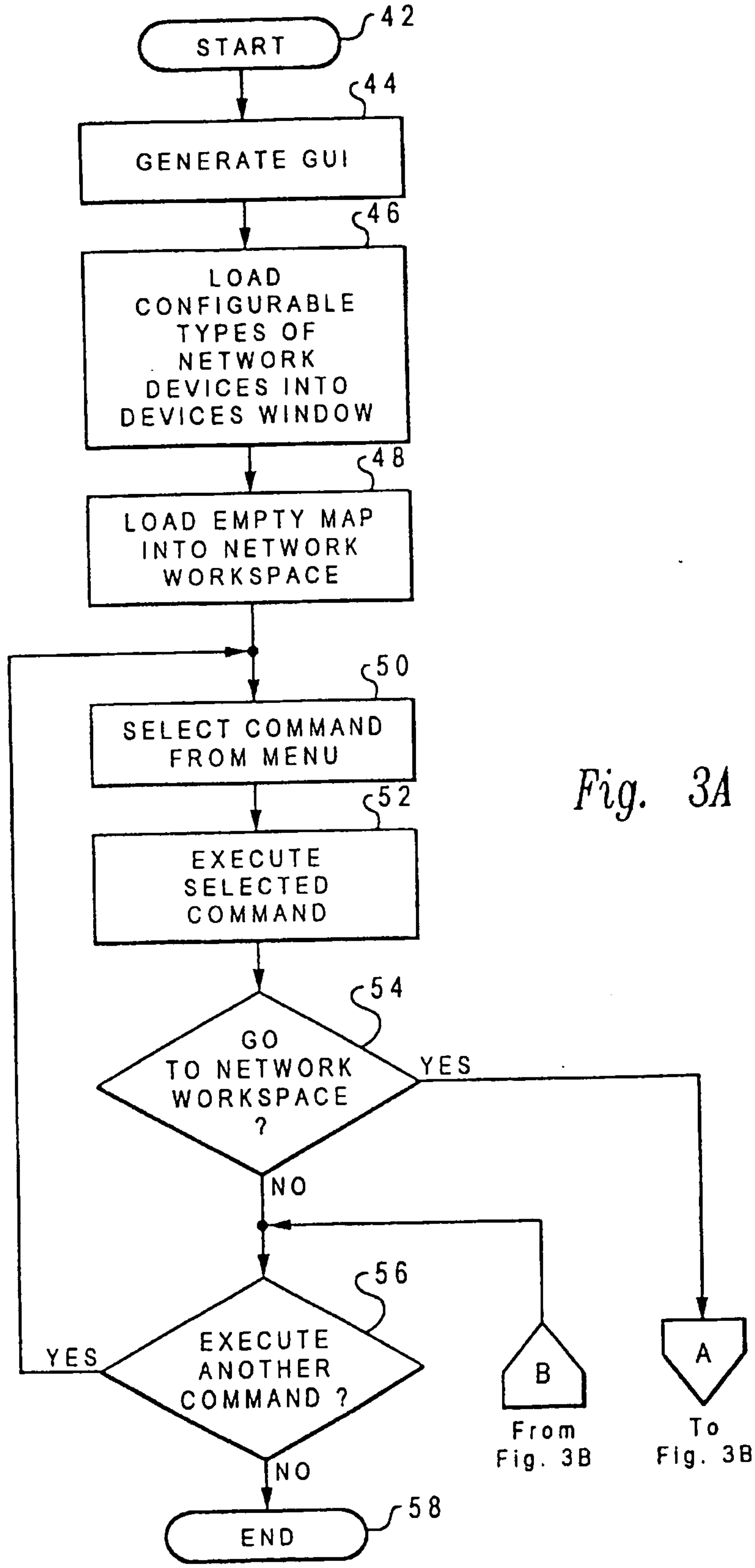

42
START
44
GENERATE GUI
46
LOAD CONFIGURABLE TYPES OF NETWORK DEVICES INTO DEVICES WINDOW
48
LOAD EMPTY MAP INTO NETWORK WORKSPACE
50
SELECT COMMAND FROM MENU
52
EXECUTE SELECTED COMMAND
54
GO TO NETWORK WORKSPACE ?
YES
NO
56
EXECUTE ANOTHER COMMAND ?
YES
NO
B
From Fig. 3B
A
To Fig. 3B
58
END

Fig. 3A

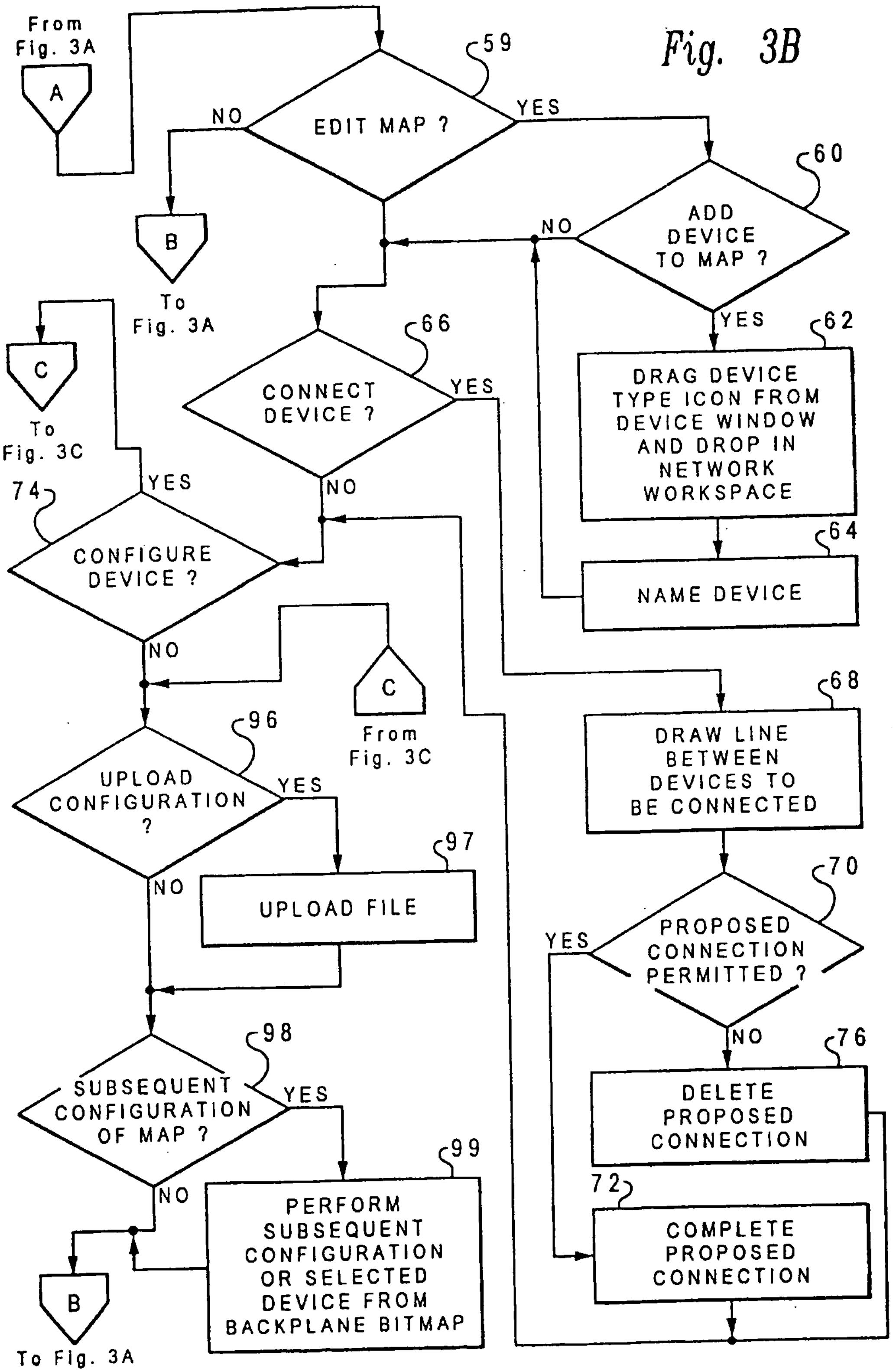

From Fig. 3A
A
59
EDIT MAP ?
NO
YES
B
To Fig. 3A
60
ADD DEVICE TO MAP ?
NO
YES
62
DRAG DEVICE TYPE ICON FROM DEVICE WINDOW AND DROP IN NETWORK WORKSPACE
64
NAME DEVICE
C
To Fig. 3C
66
CONNECT DEVICE ?
YES
NO
74
CONFIGURE DEVICE ?
YES
NO
C
From Fig. 3C
68
DRAW LINE BETWEEN DEVICES TO BE CONNECTED
96
UPLOAD CONFIGURATION ?
YES
NO
97
UPLOAD FILE
70
PROPOSED CONNECTION PERMITTED ?
YES
NO
76
DELETE PROPOSED CONNECTION
72
COMPLETE PROPOSED CONNECTION
98
SUBSEQUENT CONFIGURATION OF MAP ?
YES
NO
99
PERFORM SUBSEQUENT CONFIGURATION OR SELECTED DEVICE FROM BACKPLANE BITMAP
B
To Fig. 3A

Fig. 3B

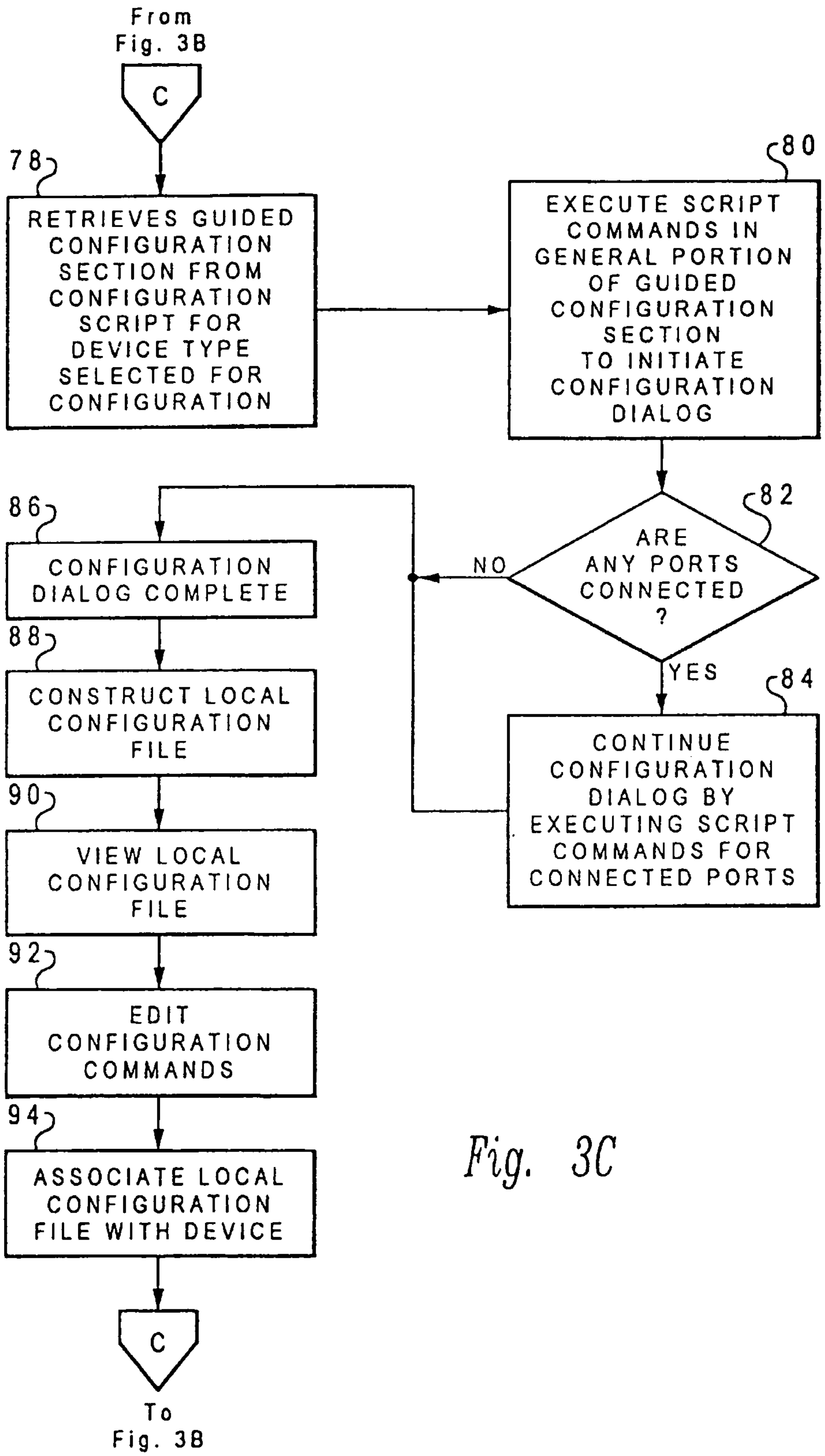

From
Fig. 3B
C
78
RETRIEVES GUIDED CONFIGURATION SECTION FROM CONFIGURATION SCRIPT FOR DEVICE TYPE SELECTED FOR CONFIGURATION
80
EXECUTE SCRIPT COMMANDS IN GENERAL PORTION OF GUIDED CONFIGURATION SECTION TO INITIATE CONFIGURATION DIALOG
82
ARE ANY PORTS CONNECTED ?
NO
YES
84
CONTINUE CONFIGURATION DIALOG BY EXECUTING SCRIPT COMMANDS FOR CONNECTED PORTS
86
CONFIGURATION DIALOG COMPLETE
88
CONSTRUCT LOCAL CONFIGURATION FILE
90
VIEW LOCAL CONFIGURATION FILE
92
EDIT CONFIGURATION COMMANDS
94
ASSOCIATE LOCAL CONFIGURATION FILE WITH DEVICE
C
To
Fig. 3B

Fig. 3C

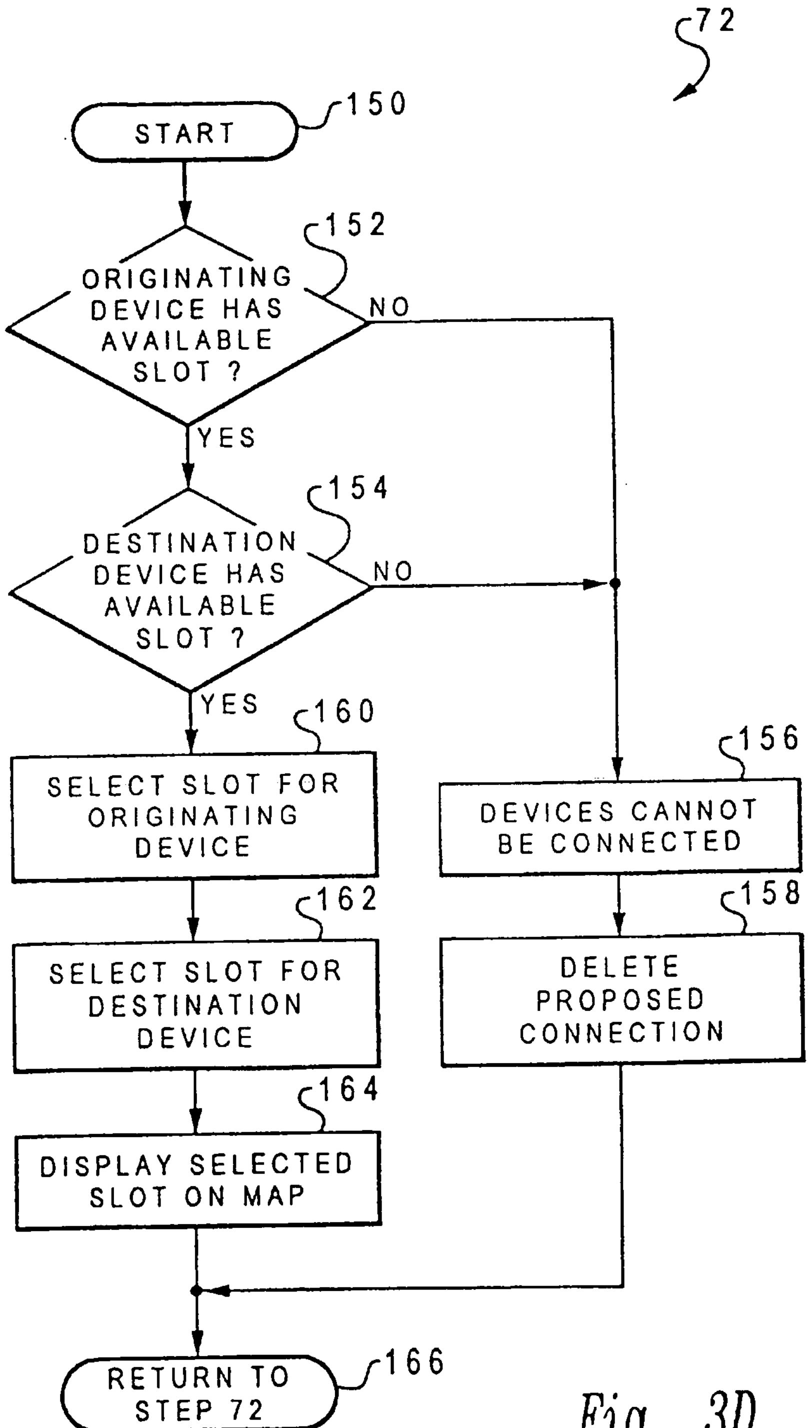

72
START
150
ORIGINATING DEVICE HAS AVAILABLE SLOT ?
152
NO
YES
DESTINATION DEVICE HAS AVAILABLE SLOT ?
154
NO
YES
SELECT SLOT FOR ORIGINATING DEVICE
160
SELECT SLOT FOR DESTINATION DEVICE
162
DISPLAY SELECTED SLOT ON MAP
164
DEVICES CANNOT BE CONNECTED
156
DELETE PROPOSED CONNECTION
158
RETURN TO STEP 72
166

Fig. 3D

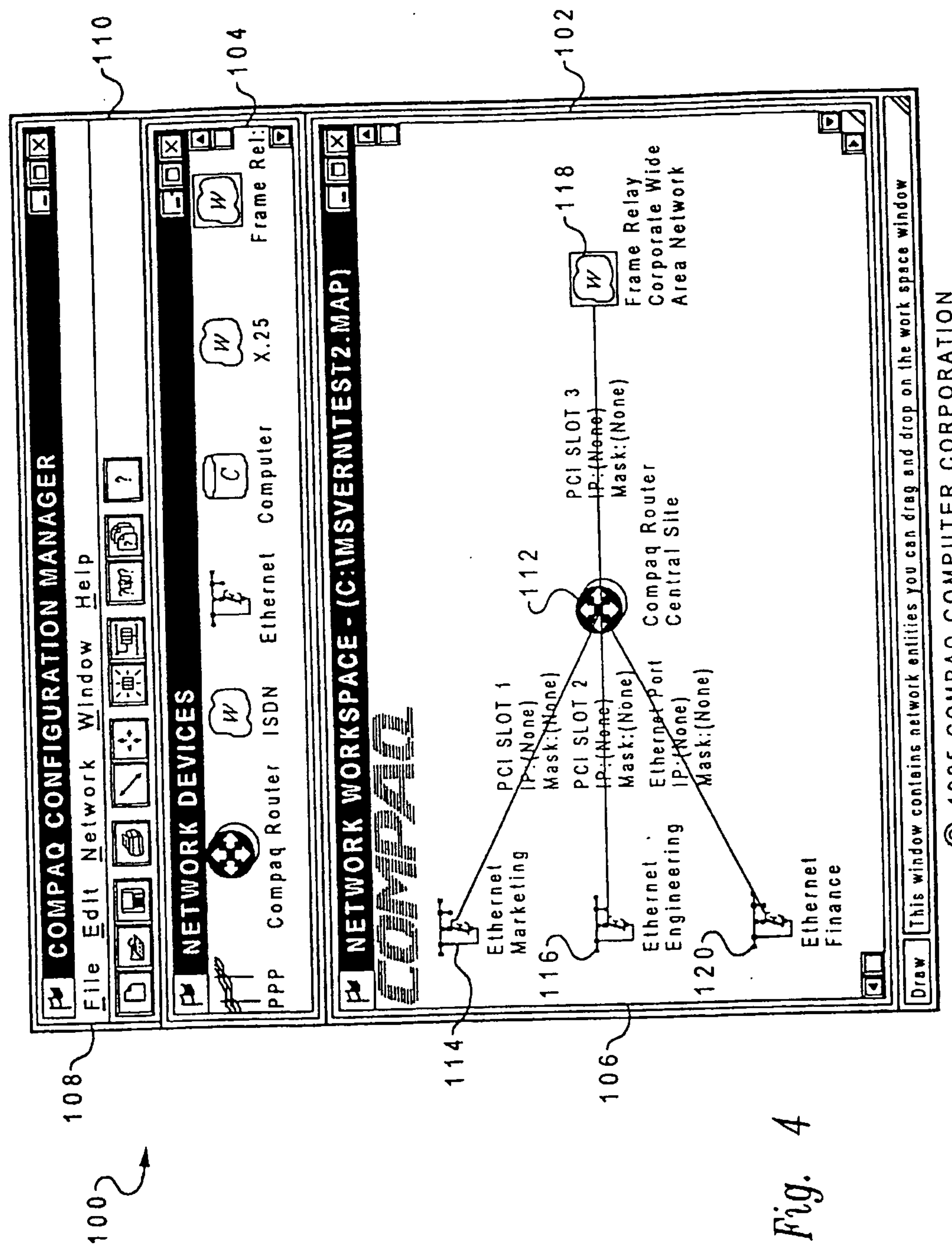

100
108
110
104
102
COMPAQ CONFIGURATION MANAGER
File Edit Network Window Help
NETWORK DEVICES
PPP
Compaq Router
ISDN
Ethernet
Computer
X.25
Frame Rel:
NETWORK WORKSPACE - (C:IMSVERNITEST2.MAP)
114
116
120
106
112
118
Ethernet Marketing
Ethernet Engineering
Ethernet Finance
PCI SLOT 1
IP:(None)
Mask:(None)
PCI SLOT 2
IP:(None)
Mask:(None)
Ethernet Port
IP:(None)
Mask:(None)
Compaq Router
Central Site
PCI SLOT 3
IP:(None)
Mask:(None)
Frame Relay
Corporate Wide
Area Network
Draw
This window contains network entities you can drag and drop on the work space window
© 1995 COMPAQ COMPUTER CORPORATION

Fig. 4

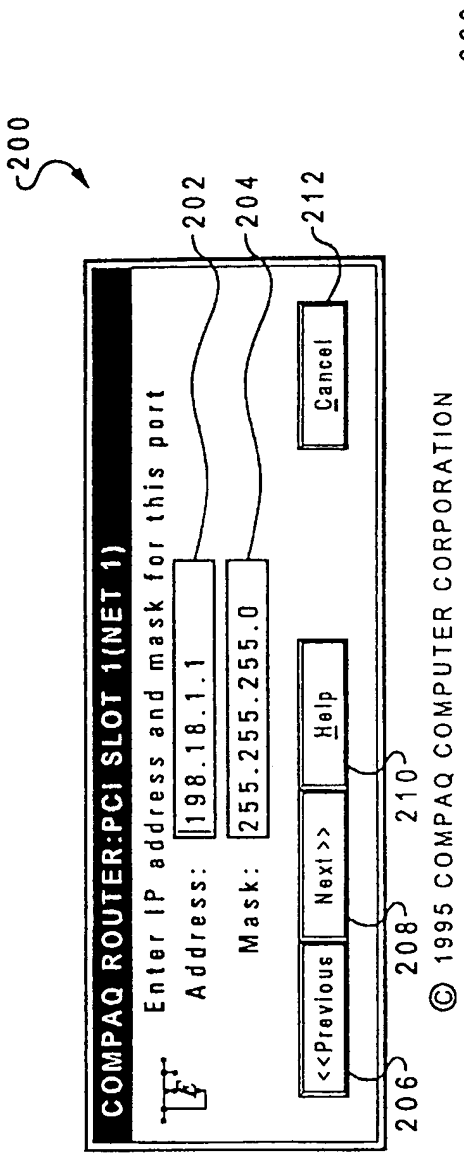

200
202
204
212
COMPAQ ROUTER:PCI SLOT 1(NET 1)
Enter IP address and mask for this port
Address: 198.18.1.1
Mask: 255.255.255.0
<<Previous
Next>>
Help
Cancel
206
208
210
© 1995 COMPAQ COMPUTER CORPORATION

Fig. 5

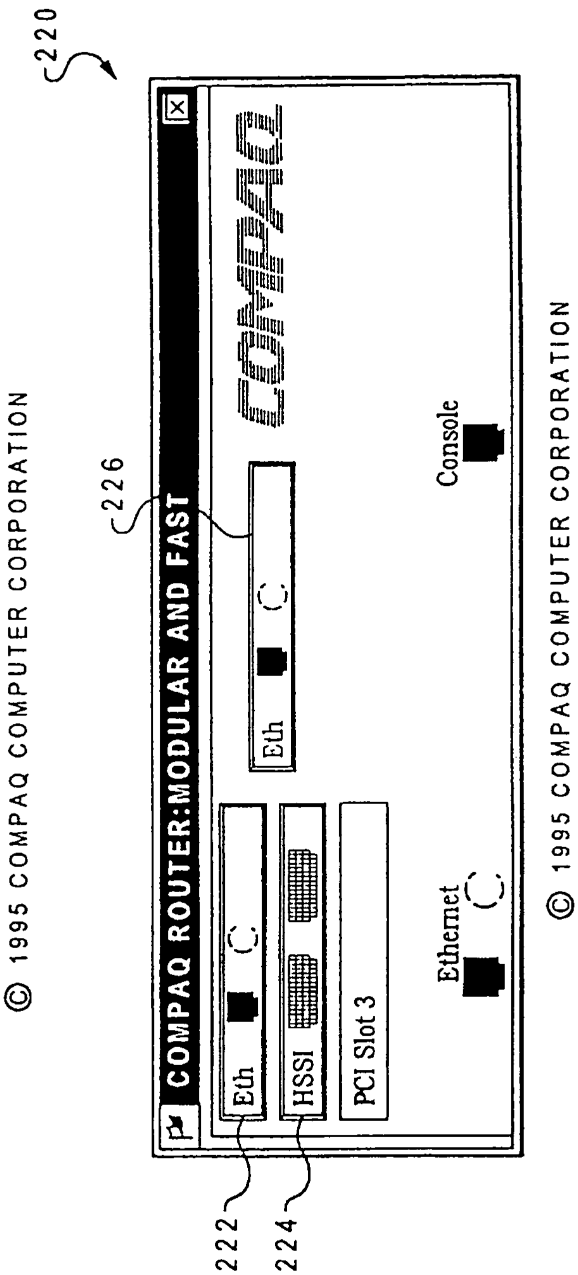

220
226
COMPAQ ROUTER:MODULAR AND FAST
COMPAQ
Eth
Eth
HSSI
PCI Slot 3
Console
Ethernet
222
224
© 1995 COMPAQ COMPUTER CORPORATION

Fig. 6

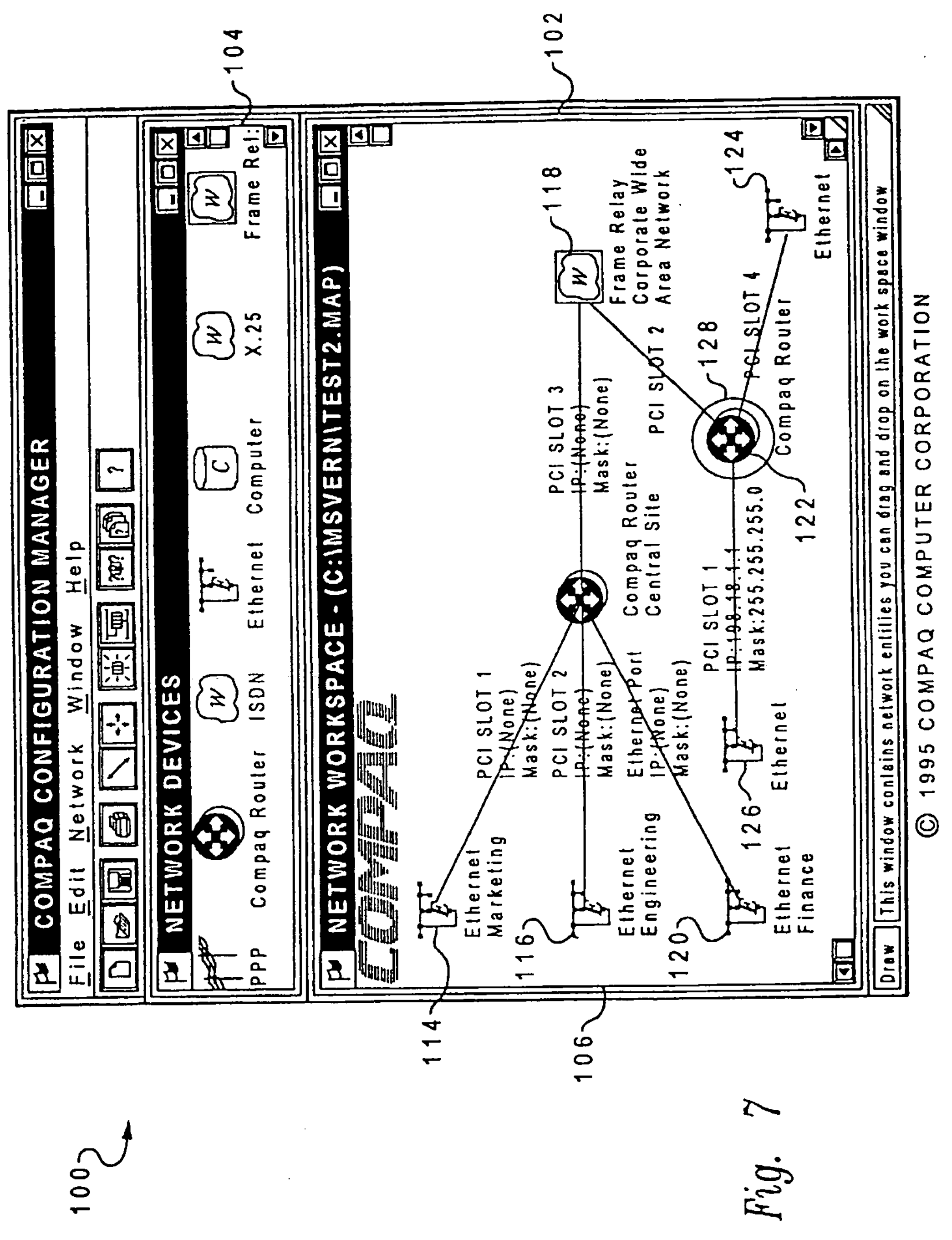

COMPAQ CONFIGURATION MANAGER
File Edit Network Window Help
NETWORK DEVICES
PPP
Compaq Router
ISDN
Ethernet
Computer
X.25
Frame Rel.
NETWORK WORKSPACE - (C:\MSVERNITEST2.MAP)
Ethernet
Marketing
Ethernet
Engineering
Ethernet
Finance
PCI SLOT 1
IP:(None)
Mask:(None)
PCI SLOT 2
IP:(None)
Mask:(None)
Ethernet Port
IP:(None)
Mask:(None)
Compaq Router
Central Site
PCI SLOT 3
IP:(None)
Mask:(None)
Ethernet
PCI SLOT 1
IP:198.18.1.1
Mask:255.255.255.0
PCI SLOT 2
PCI SLOT 4
Compaq Router
Frame Relay
Corporate Wide
Area Network
Ethernet
Draw
This window contains network entities you can drag and drop on the work space window
© 1995 COMPAQ COMPUTER CORPORATION
100
102
104
106
114
116
118
120
122
124
126
128

Fig. 7

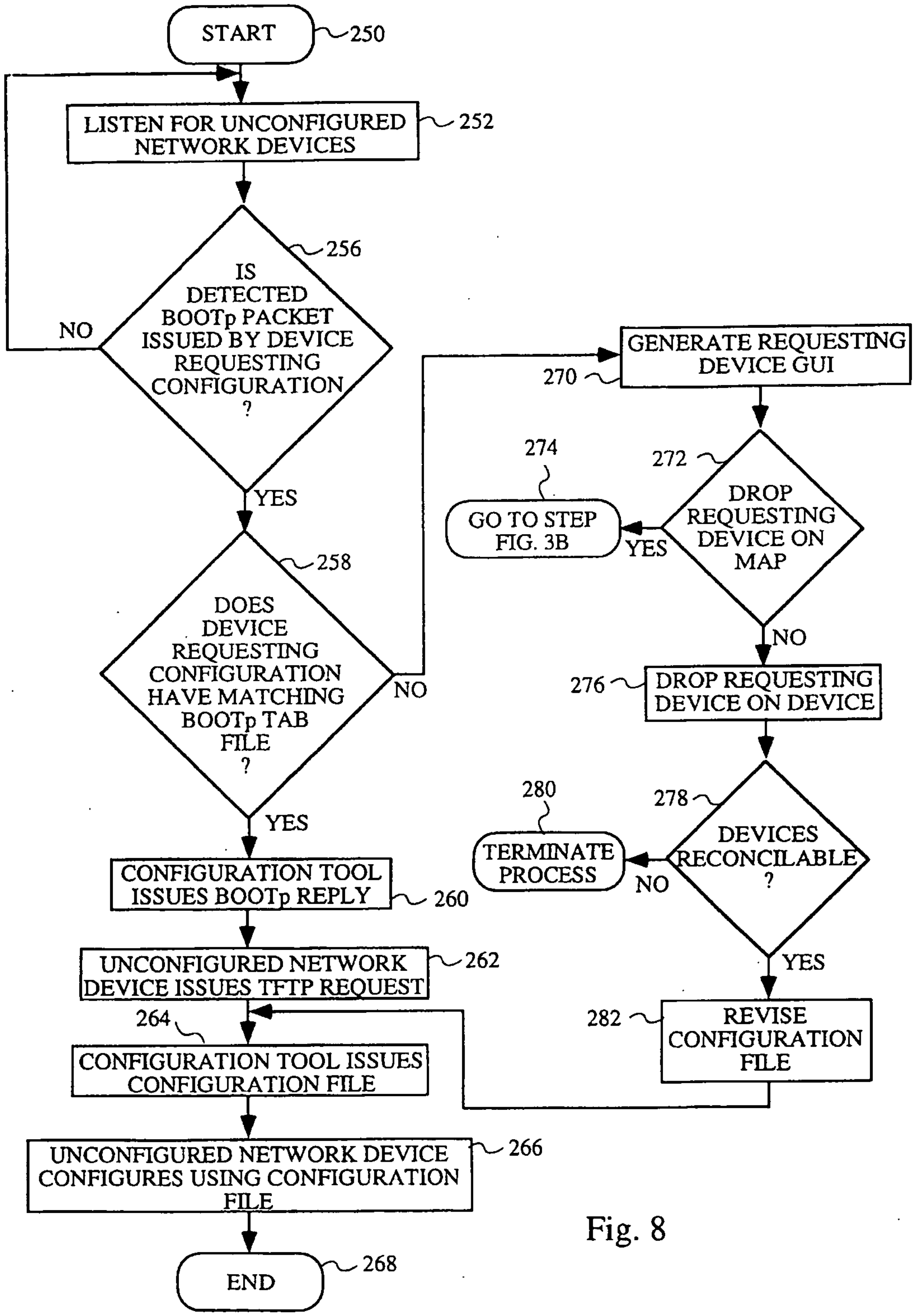
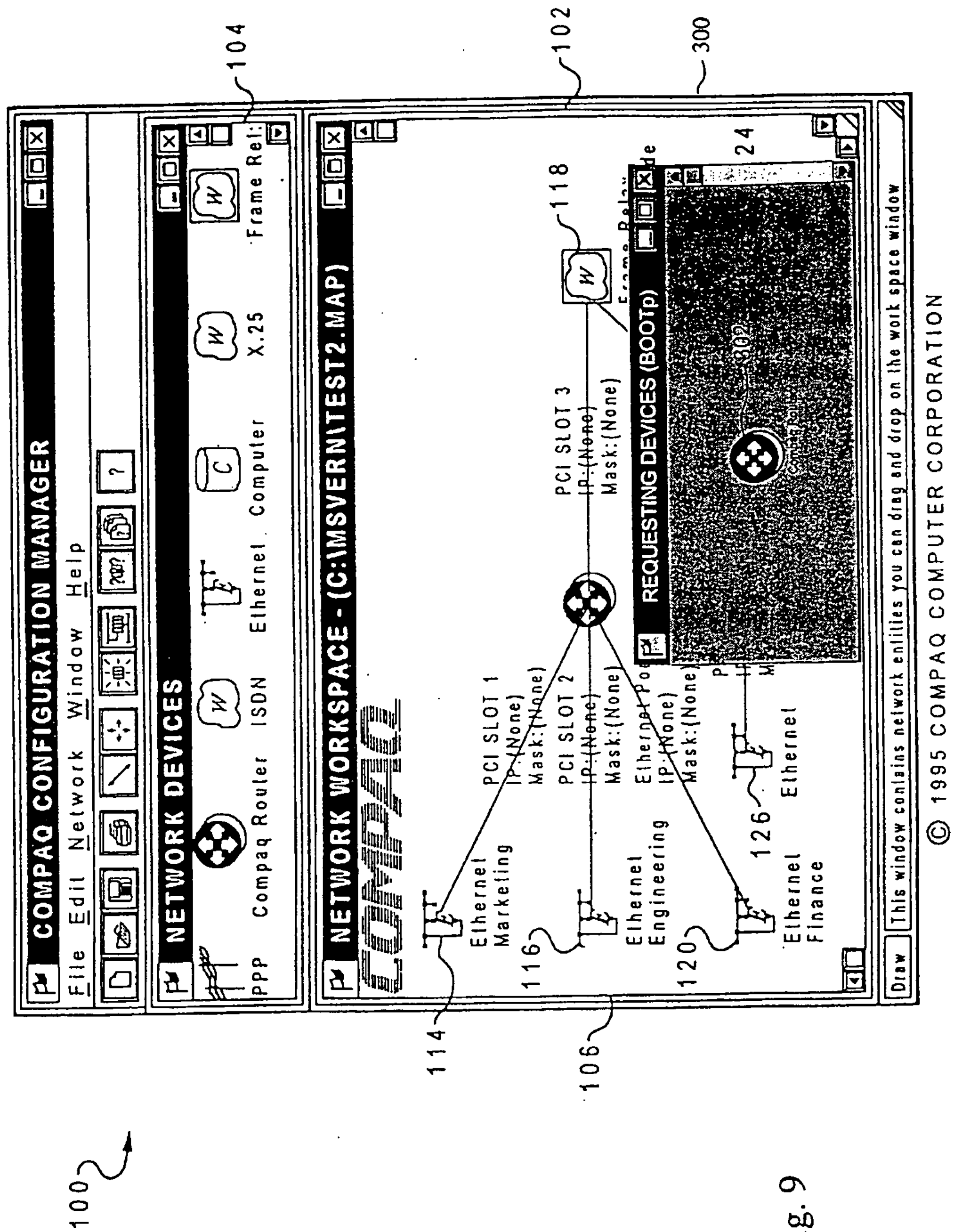

START
250
LISTEN FOR UNCONFIGURED NETWORK DEVICES
252
IS DETECTED BOOTp PACKET ISSUED BY DEVICE REQUESTING CONFIGURATION ?
256
NO
YES
DOES DEVICE REQUESTING CONFIGURATION HAVE MATCHING BOOTp TAB FILE ?
258
NO
YES
CONFIGURATION TOOL ISSUES BOOTp REPLY
260
UNCONFIGURED NETWORK DEVICE ISSUES TFTP REQUEST
262
264
CONFIGURATION TOOL ISSUES CONFIGURATION FILE
UNCONFIGURED NETWORK DEVICE CONFIGURES USING CONFIGURATION FILE
266
END
268
GENERATE REQUESTING DEVICE GUI
270
DROP REQUESTING DEVICE ON MAP
272
YES
GO TO STEP FIG. 3B
274
NO
DROP REQUESTING DEVICE ON DEVICE
276
DEVICES RECONCILABLE ?
278
NO
TERMINATE PROCESS
280
YES
REVISE CONFIGURATION FILE
282

Fig. 8



COMPAQ CONFIGURATION MANAGER
File Edit Network Window Help
NETWORK DEVICES
PPP
Compaq Router
ISDN
Ethernet
Computer
X.25
Frame Rel:
NETWORK WORKSPACE - (C:\MSVERN\TEST2.MAP)
PCI SLOT 1
IP:(None)
Mask:(None)
PCI SLOT 2
IP:(None)
Mask:(None)
PCI SLOT 3
IP:(None)
Mask:(None)
Ethernet
Marketing
Ethernet
Engineering
Ethernet
Finance
Ethernet
REQUESTING DEVICES (BOOTp)
Draw
This window contains network entities you can drag and drop on the work space window
© 1995 COMPAQ COMPUTER CORPORATION
100
102
104
106
114
116
118
120
124
126
300
302

Fig. 9

CONFIGURATION MANAGER FOR NETWORK DEVICES AND AN ASSOCIATED METHOD FOR PROVIDING CONFIGURATION INFORMATION THERETO

CROSS REFERENCE TO RELATED APPLICATION

This application is related to co-pending U.S. patent application Ser. No. 08/603,061, filed on even date herewith, entitled METHOD AND APPARATUS FOR GUIDED CONFIGURATION OF UNCONFIGURED NETWORK AND INTERNETWORK DEVICES", assigned to the Assignee of the present application and hereby incorporated by reference as if reproduced in its entirety.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This application generally relates to computer networks and internetworks and, more particularly, to a configuration manager which, from a central location, provides configuration information to remote devices included in a computer network or internetwork.

2. Description of Related Art

Generally speaking, a network is a collection of user devices, generally classified as data terminal equipment (or "DTE"), interconnected for bi-directional exchanges of information. For example, visual displays, computer systems and office workstations are all electronic devices classified as DTEs. A local area network (or "LAN") is an interconnection of plural computer systems distributed around a single site. A wide area network (or "WAN") is an interconnection of plural computer systems located at different sites. Traditionally, computer systems have used modems to connect to a WAN via the public switched telephone network (or "PSTN") or public switched data network (or "PSDN"). In recent years, WANs which utilize integrated services digital networks (or "ISDNs"), which enable data to be transmitted without modems, to interconnect computer systems have become more common. Finally, an internetwork is a collection of networks interconnected by a WAN.

Devices are initially unconfigured when delivered by the factory. Configuration is a process during which the hardware and software of an unconfigured device is organized and interconnected so that the configured device will be able to perform the tasks desired thereof. As is well appreciated in the art, the wide variety of devices which may be installed on a network, as well as the variety of networks which may be connected to form an internetwork, makes the configuration of networks and internetworks a difficult task which requires highly detailed technical knowledge of the various networks, the protocols used to link with the various networks and the devices to be installed thereon. Thus, configuration of network devices is often one of the most daunting tasks facing a network administrator, particularly for those in charge of small and medium size networks have between 100 and 1,000 nodes. While such networks are relatively complex, their administrators often have only minimal training in internetworking administration and may be unfamiliar with routing technology and/or WAN technology.

For example, data link protocols are used to control access to networks. A LAN typically uses the logical link control (or "LLC") subclass of the high-level data link control (or "HDLC") protocol as its data link protocol. However, an X.25-type packet-switching WAN uses link access procedure, balanced (or "LAPB"), a protocol based on HDLC, as its data link protocol. The data link protocol for an ISDN-type WAN, on the other hand, may either be a connection-orientated protocol known as frame switching or a connectionless protocol known as frame relay.

Even when configuration information is available, further complications problems arise when transporting the configuration information to a network device. For example, in order to communicate with a remotely located network device, a network administrator needs to know where to send the information. However, that knowledge typically resides in the configuration information for the network device. Thus, the network administrator is constrained as to what information may be delivered to the network device until after the device is configured but, until the device is configured, much of the configuration information is undeliverable. While techniques for transporting information to an unconfigured device using a limited amount of configuration information exist, inconsistencies in such information often complicate the task of transferring information using such techniques.

Thus, it can be readily seen from the foregoing that it would be desirable to simplify the task of configuring a remotely located network device. It is, therefore, the object of this invention to provide a configuration manager and an associated method of configuring a remote network device from a central location.

SUMMARY OF THE INVENTION

In one embodiment, the present invention is of a configuration manager for configuring a network device remotely coupled thereto. The configuration manager includes a configuration script stored in a memory subsystem of a computer system and first and second software modules respectively executable by a processor subsystem of the computer system. The configuration script contains a series of executable instructions for constructing a configuration file and a bootptab file for a first specified type of network device. By executing the instructions contained in the configuration script, the first software module may construct a configuration file suitable for upload to a network device and a bootptab file suitable for identifying the network device. Configuration requests issued by the network device are processed by the second software module by identifying the requesting network device using the constructed bootptab file and configuring the requesting network device by uploading the constructed configuration file thereto.

In one aspect of this embodiment of the invention, the configuration script includes a first section which contains a series of configuration commands which generate requests for information such that information received by the first software module in response to the requests for information is used to construct the configuration and bootptab files. In another aspect of this embodiment of the invention, the configuration script includes a second section which contains a set of connection rules for connecting the first specified type of network device to at least one other specified type of network device.

In a related aspect thereof, the second section of the configuration script includes a first portion which uniquely identifies the network device and a second portion which identifies devices installed in the network device. In another related aspect thereof, the first section of the configuration script includes a first portion which corresponds to each of the at least one other specified type of network device specified in the connection rules contained in the second section of the configuration script. Each such portion contains a subset of the series of configuration commands contained in the first section of the configuration script and each such subset of configuration commands are executed only if the network device for which the configuration file is being constructed is connected to a network device of the other specified type of network device.

In another embodiment, the present invention is of a computer-implemented method for configuring a remotely located network device. A request for configuration issued by a network device is detected. If a previously constructed configuration file corresponds to the network device issuing the request for configuration, a reply which identifies the configuration file is transmitted to the network device. The configuration file is then transmitted to the network device in response to a request for the identified configuration file. The configuration file is constructed using a configuration script containing a series of executable instructions for constructing a configuration file for a first specified type of network device is provided. The configuration file is then constructed by executing the series of instructions contained in the configuration script. In one aspect thereof, the configuration script includes a first section containing a series of configuration commands. Requests for information are issued by executing the series of configuration commands contained in the first section of the configuration script and information received in response to the requests for information is used to construct the configuration file. The information may also be used to construct a bootptab file which, in addition to the configuration file, contains a unique identifier for the network device.

In another aspect of this embodiment of the invention, a determination of whether the configuration file corresponds to the network device issuing the request for configuration is accomplished by determining if the network device issuing the request for configuration has an identification code which matches an identification code contained in the bootptab file and determining if devices installed in the network device issuing the request for configuration match the installed devices identified in the bootptab file.

In yet another aspect of this embodiment of the invention, the provided configuration script may also include a second section containing a set of connection rules for connecting the first specified type of network device to at least one other specified type of network device. Within the second section, a first portion corresponding to each of the at least one other specified type of network device specified in the connection rules contained in the second section of the configuration script may also be provided. Each first portion contains a subset of the series of configuration commands which are executed only if the network device for which the configuration file is being constructed is connected to a network device of the other specified type of network device.

BRIEF DESCRIPTION OF THE DRAWING

The present invention may be better understood, and its numerous objects, features and advantages will become apparent to those skilled in the art by reference to the accompanying drawing, in which:

FIG. 1A is a simplified block diagram of a computer system on which a network device configuration tool may be installed;

FIG. 1B is a block diagram of a network device configuration tool constructed in accordance with the teachings of the present invention;

FIG. 2A is a block diagram of a configuration scripts portion of the network device configuration tool of FIG. 1;

FIG. 2B is an expanded block diagram of an attributes section of a configuration script of FIG. 2A;

FIG. 2C is an expanded block diagram of a bitmap section of a configuration script of FIG. 2A;

FIG. 2D is an expanded block diagram of a bitmap menu section of a configuration script of FIG. 2A;

FIG. 2E is an expanded block diagram of a guided configuration section of a configuration script of FIG. 2A;

FIG. 3A is a flow chart of a method for guiding configuration of a network device in accordance with the teachings of the present invention;

FIG. 3B is a flow chart of a map edit section of the flow chart of FIG. 3A;

FIG. 3C is a flow chart of a guided configuration subsection of the flow chart of FIG. 3B;

FIG. 3D is a flow chart of a method for determining whether a pair of network devices are connectable;

FIG. 4 illustrates a configuration manager GUI for constructing a map of configured network devices with a preconstructed network configuration map in a network workspace portion thereof;

FIG. 5 illustrates an exemplary guided configuration GUI for constructing a configuration script for a network device;

FIG. 6 illustrates a backplane bitmap for a configured network device;

FIG. 7 illustrates the configuration manager GUI of FIG. 4 with a preconstructed network configuration map modified to include newly added and configured devices thereon;

FIG. 8 is a flowchart of a method of configuring a remote network device in accordance with another aspect of the present invention; and

FIG. 9 illustrates a pop-up bootP GUI in which an unconfigured network device is requesting configuration information.

DETAILED DESCRIPTION

Referring first to FIG. 1A, a computer system **2** suitable for installing a network device configuration tool thereon may now be seen. The computer system **2** is comprised of a processor subsystem **4**, for example, a type P6 Pentium processor manufactured by Intel Corporation of Santa Clara, Calif., coupled to a memory subsystem **6**, for example, a hard drive or other auxiliary memory device capable of storing large amounts of data infrequently used by the processor subsystem **4**, by a system bus **8**, preferably, a 32-bit wide peripheral connection interface (or "PCI") bus. Also coupled to the system bus **8** is a user interface **9**. Commonly, the user interface is comprised of three peripheral devices—a video display, a keyboard and a pointing device.

Referring now to FIG. 1B, a network device configuration tool **10** constructed in accordance with the teachings of the present invention will now be described in greater detail. The network device configuration tool **10** is graphical user interface (or "GUI") based software launchable from a suitable platform installed on the computer system **2**. For

example, Windows 95 and Windows NT 3.51, both manufactured by Microsoft of Redmond, Wash., are suitable platforms from which the network device configuration tool **10** may be launched.

In its broadest sense, the network device configuration tool **10** provides a GUI in which the so-called "drag and drop" process is used to construct a network configuration map comprised of a series of interconnected network devices and/or network entities, for example, a LAN, WAN or other network, from a combination of user inputs, network configuration maps, configuration scripts and local configuration files.

In constructing the network configuration map, a series of local configuration files are constructed for the network devices and appended to the network configuration map. The local configuration files contain information, for example, internet protocol (or "IP") address, default gateway, router name and simplified network management protocol (or "SNMP") community strings, necessary for the network device, for example, a router, to properly communicate on the network.

For each network device for which a local configuration file has been constructed, the network device configuration tool **10** may also construct a network device configuration file suitable for export to the network device itself. In this manner, remote configuration of network devices is enabled.

As shown in FIG. **1B**, the network device configuration tool **10** may be representatively illustrated as being comprised of two software modules, map editor **14** and configuration guide **18**, both of which are executable by the processor subsystem **4**, which retrieve data and programming instructions from various locations within the memory subsystem **8** of the computer system **2** on which the network device configuration tool **10** is installed.

The data and programming instruction are stored in the memory subsystem **6** as a series of files which may be selectively accessed by the map editor **14** and/or the configuration guide **18**. Files which are accessible to the map editor **14** and/or the configuration guide **18** are configuration scripts **12**, map files **16**, local configuration files **20** and network configuration files **22**. The configuration scripts **12** identify the types of network devices and network entities which may be placed on the network configuration map and interconnected with other network entities and network devices. The configuration scripts **12** also identify the network devices which are configurable by the network device configuration tool **10** and contain information necessary to construct configuration files for those network devices. If a particular network device does not have a configuration script, a configuration file cannot be constructed by the network device configuration tool **10**. The map files **16** contain a series of network configuration maps, each comprised of a series of interconnected network devices and network entities, constructed using the network device configuration tool **10**. The local configuration files **20** contain information which, if uploaded to the corresponding network device **26**, would enable configuration of that device. If local configuration files **20** are constructed for the network devices illustrated on the network configuration map(s) **16** produced using the network device configuration tool **10**, such local configuration files **20** are associated with the corresponding network device such that they may be directly accessed from the network configuration maps **16**.

The network configuration files **22** are similar in content to the local configuration files **20** except that the files have been formatted for upload to a network device **26** coupled to the configuration tool in a manner to be more fully described below. Broadly speaking, a local configuration file **20** is modified for upload to the corresponding network device **26** by formatting the local file into the appropriate IP address for the target network device **26**. Finally, the network device configuration tool **10** includes a reverse parser **24** coupled to the local configuration files **20** and the network configuration files **22**. The reverse parser **24** is used to construct a local configuration file **20** from a network configuration file **22** downloaded to the network configuration tool **10** by the network device **26**.

It is contemplated that the network device configuration tool **10** would be installed in the computer system **2** operated by a network administrator and that plural network devices **26** and other network entities, only one of which is shown in FIG. 1B for ease of illustration, would be coupled to the network device configuration tool **10**. Utilizing the network device configuration tool **10**, the network administrator may build a representative network configuration map for the network. The network administrator may then configure remotely located network devices by uploading configuration files constructed during the process of building the network configuration map to the devices. Thus, by using the network configuration tool, the network administrator can, from a central location, design a suitable configuration network and then configure any number of remotely located devices included in the network.

The network device configuration tool **10** is coupled to the network device **26** by an asynchronous interface **28** and a boot protocol (or "bootp")/trivial file transfer protocol (or "TFTP") manager **30**. Under the control of an asynchronous manager (not shown), a software process within the processor subsystem **4**, the asynchronous interface **28** is used to exchange configuration information, for example, a network configuration file **20**, by either an in-band transfer via in-band connection **29*a***, for example, via telnet, or by an out-of-band transfer via out-of-band connection **29*b***, for example, via modem. Additionally, the bootp/TFTP manager **30**, another software process within the processor subsystem **4**, controls the exchange of bootp and TFTP messages between the network device configuration tool **10** and the network device **26**. Generally, a bootp exchange is used to transfer raw address and other basic information so that a TFTP exchange may then be used to transfer configuration information. The bootp/TFTP manager **30** also controls accesses to bootptab files **32**.

As will be more fully described with respect to FIG. **3**, below, the configuration scripts **12** are used to direct map editor **14** and configuration guide **18** in a guided configuration of a selected network device **26** by guiding in the construction of a configuration file for the device. Accordingly, turning momentarily to FIG. **2A**, the configuration scripts **12** used to guide the configuration of a selected network device **26** will now be described in greater detail. As may now be seen, the configuration scripts **12** are comprised of a series of separate scripts **12-1** through **12-N**, one for each type of device which may configured by the configuration tool **12**. Each script **12-1** through **12-N** is comprised of an attributes section **34**, a bitmap section **36**, a bitmap menu section **38** and a guided configuration section **40**. Each of these sections **34, 36, 38** and **40** is a selectively executable set of commands which may be used during configuration of a device of the type corresponding to a particular script **12-1** through **12-N**.

Turning now to FIG. **2B**, the attributes section **34** is comprised of an icon portion **34*a***, a network entity portion **34*b***, a description portion **34*c*** and a series of connection

portions **34*d*-1** through **34*d*-N**. A valid icon filename identifying the graphical icon to be associated with the device type corresponding to the configuration script **12-N** is contained in the icon portion **34*a***. As will be more fully described below, this icon will appear in a device window of a configuration GUI and can be dragged onto a network workspace to add a device of that type to a network configuration map. The network entity portion **34*b*** provides a unique name for the type of device and appears in the device window under the icon. The description portion **34*c*** defines a default description for the device which pre-populates the dialog box when a device type is dragged onto the network workspace. Finally, the connection portions **34*d*-1** though **34*d*-N** provides connection statements for the device type. Specifically, a connection portion **34*d*** will be provided for port, modular slot or other type of connection interface for the device type. Each connection statement will include a physical name for the port or other type of connection interface and the network entity names for all other types of devices which may be connected to the port. For example, if the network device was a modular router having 4 PCI slots, each connectable to ethernet, X.25, frame relay, PPP and IDSN type entities, and an ethernet port connectable to an ethernet entity, the attributes section **34** could be as set forth in the following code:

```
:ATTRIBUTES
ICON ROUTER.ICO
NETENTITY "Compaq Router"
DESCRIPTION "Modular and Fast"
CONNECT   "PCI SLOT 1"  "Ethernet"  "X.25"   "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "PCI SLOT 2"  "Ethernet"  "X.25"   "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "PCI SLOT 3"  "Ethernet"  "X.25"   "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "PCI SLOT 4"  "Ethernet"  "X.25"   "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "Ethernet Port" "Ethernet"
© 1995 Compaq Computer Corporation
```

Turning next to FIG. 2C, the bitmap section **36** defines the "drill down" bitmap which is presented to the network administrator upon requesting subsequent configuration of a configured network device. The bitmap section **36** also defines any necessary overlay bitmaps as well as provides the locations of "hot spots" on the bitmap. The bitmap is a graphical representation of the backplane of the configured device which provides connection information for the ports thereof. "Hot spots" on the bitmap are paths to additional information related to the connected ports for the configured network device.

Bitmap file portion **36*a*** names a valid window bitmap format file which will be displayed in its own window when the network administrator double clicks on a configured network device. For each connected port of the configured network device, the bitmap section **36** will also include a location port portion **36*b*-1** through **36*b*-N** which provides the location of the hot spot for the connected port on the bitmap. Finally, the bitmap section includes an overlay device bitmap file **36*c*-1** through **36*c*-x** for each type of network device or entity which is connectable to the configured network device. Then, if the configured device is connected to that particular network entity, the network entity can be represented on the bitmap.

For example, if the bitmap **36** is comprised of a bitmap file **36*a***, port locations **36*b*-1** and **36*b*-2** and overlay device file **36*c*-1** as set forth in the sample code below:

```
BITMAP "router.bmp"
LOCATE "Slot 1" 20 40
LOCATE "Port 1" 50 90 70 120
OVERLAY "Ethernet" "TLAN.BMP"
© 1995 Compaq Computer Corporation
```

The bitmap **36** will include a representation of an ethernet-type network entity stored at TLAN.BMP drawn on top of the representation of a backplane of a router stored at ROUTER.BMP at coordinates **20, 40** if the "Ethernet"-type network entity is plugged into "Slot 1".

The bitmap menu section **38** defines a menu hierarchy presented to the user for hot spots, for example, connected slots, on the bitmap and the executable commands for each item included in a command menu. The command menu is displayed when the network administrator clicks on a connected slot on the bitmap. The bitmap menu section **38** is subdivided into network entity command sections **38*a*-1** through **38*a*-x**. Specifically, for each network entity for which connection to the device is allowed, a corresponding network entity command section is provided such that, if that network entity is connected to the device, the commands defined in the section will be displayed to the network administrator for selective execution thereof.

The guided configuration section **40** defines the GUIs used to guide a user through configuration of a device and controls the configuration file to be constructed using user responses to the GUIs. As illustrated in FIG. 2D, the guided configuration section **40** is subdivided into a general script command portion **40*a*** and a port script command portion **40*b*-1** through **40*b*-N** for each port to which the device is connectable. A guided configuration script for a Cisco 2514 router is set forth in Appendix A by way of example and will be described in greater detail with respect to FIG. 3-D, below.

Returning now to FIG. 1B, the network device configuration tool **10** will now be described in greater detail. Generally, the map editor **14** controls the generation of a map of a network configuration while delegating the task of configuring unconfigured devices placed on the network configuration map to the configuration guide **18**. Upon initiation of the configuration process, the map editor **14** selectively retrieves a map file **16**, or creates a blank map, for editing. To add a device of a selected type to the network configuration map, the map editor **14** retrieves the corresponding configuration script **12-N** from the configuration scripts **12** and, using the information contained in the retrieved configuration script **12-N**, places an unconfigured device of the selected type on the network configuration map and appends a name for the device to the map. The map editor **14** performs all operations in which editing of the network configuration map is proposed. For example, if a connection between two devices placed on the network configuration map is proposed, the map editor **14** reviews the configuration scripts **12** for the devices and, if a connection between the two devices is permitted, the map editor **14** completes the proposed connection and appends the connection information to the network configuration map.

If a request to configure a device placed on the network configuration map is received, the map editor **14** transfers the name and connection information for the device to the configuration guide **18** and instructs the configuration guide **18** to perform the requested configuration task. For example, if configuration of a network device is requested, the configuration guide **18** will retrieve the configuration script **12-N** for that type of network device and execute the

instructions contained in the guided configuration section **40** thereof. Using the information provided by the configuration script **12**, the map editor **14** and input provided by the network administrator in response to execution of the instructions contained in the guided configuration section **40**, the configuration guide **18** builds a local configuration file, associated with the device, for use by the network administrator and a corresponding network configuration file suitable for upload to the network device to enable configuration of the network device.

Referring next to FIG. 3A, the method for guiding configuration of a network device by constructing a configuration file for the network device which is the subject of the present invention shall now be described in greater detail. It should be clearly understood, however, that the illustrated order of steps is purely exemplary and should not be construed as limiting the scope of the invention. The method commences at step **42** by launching the network device configuration tool **10** from a platform such as Windows '95 by selecting an icon previously designated as providing a path to the network device configuration tool **10**.

Proceeding to step **44**, once launched, the network device configuration tool **10** generates a configuration manager GUI **100** (see FIG. 4) which provides a network workspace **102** and a device window **104**. In the network workspace **102**, a map comprised of any number of interconnected network devices, each having a configuration tied thereto, may be produced. The device window **104**, on the other hand, displays all of the types of network devices which may be placed on the network workspace **102**. Continuing on to step **46**, for each type of network device for which a configuration script **12**-N has been prepared and stored in the memory subsystem **6**, the network device configuration tool **10** places an icon representative of the network device type in the device type window **104** to indicate to the user which types of network devices are configurable by the network device configuration tool **10**. For example, the device window **104** illustrated in FIG. 4 includes icons representative of a PPP link, a vendor specific modular router, an ISDN-type WAN, an Ethernet-type LAN, a non-vendor specific computer subsystem, an X.25-type packet-switching WAN, and an ISDN-type WAN which subscribes to frame relay-mode service. At step **48**, the network device configuration tool **10** loads a blank map into the network workspace **102**. At this stage, the network device configuration tool **10** has completed loading the configuration manager GUI **100** and is ready to execute selected commands in response to inputs received from the network administrator via the user interface **9**.

Proceeding on to step **50**, the network administrator selects a command, either from one of the pull-down menus listed on the pull-down menu bar **108** or by depressing a command button displayed on command button bar **110**. The menus displayed on the pull-down menu bar **108** are "file", "edit", "network", "window" and "help". By selecting one of these menus, a series of commands, each of which relates to the selected menu, are displayed. Available file commands are "new", "open", "save", "save as", "print", "print setup" and "exit". The new command clears the network workspace **102** of any network configuration map placed thereon. The open command allows the network administrator to select a network configuration map to be placed on the network workspace **102**. The save and save as commands stores the map placed on the network workspace **102** to the memory subsystem **6**. The print command prints the network configuration map placed on network workspace **102**. The print setup command displays the printer configuration for the computer system **10**. The exit command closes the network configuration tool.

Commands available under the edit menu are "draw mode", "move mode", "workspace properties", "edit device", "view/configure device", "delete device", "all ports connected configuration", "update configuration", "retrieve configuration", "associate configuration", "telnet to this device". The draw mode command allows the network administrator to draw connections between devices displayed on the network workspace **102**. The workspace properties command is, in fact, a second pull-down menu which allows the network administrator to tailor the map placed in the network workspace **102**. Commands available under the workspace properties menu are "view entity name", "view entity description", "view entity connections", "view ip addresses", "view ipx addresses", all of which add the listed information to the display of each device on the map, and the "snap to grid" and "view grid", both of which orientate the map to a grid.

The edit device command accesses the configuration information associated with a selected network device. The view/configure command displays a view of the backplane of a selected configured network device or, if the selected network device is unconfigured, defaults to the configuration dialog set forth in greater detail below. The delete device command removes a selected network device or entity from the network workspace. The all ports configured, update configuration provides access to a selected device's configuration file. The retrieve configuration file allows the network administrator to directly access a configuration file stored in the memory subsystem **6** while the associate configuration command permits the network administrator to append a configuration file to a device. The telnet to the device command initiates an in-band transfer of configuration information from the network device configuration tool **10** to the network device **26**.

Commands available under the network menu are "bootptab maintenance", "enable bootp server", "disable bootp server", "enable TFTP server", "disable TFTP server" and "view network activity log". All of these commands are relate to the exchange of configuration information between the network device configuration tool **10** and the network device **26**. More specifically, the bootptab maintenance command enables the network administrator to review previously constructed bootptab files **32**. The enable/disable bootp server commands control the operation of the computer system **2** on which the network device configuration tool **10** operates as a bootp server, i.e. is capable of sending and/or receiving bootp messages via the bootp/TFTP manager **30**. When enabled as a bootp server, the computer system **2** listens for bootp requests placed on the network by devices requesting configuration information. The enable/disable TFTP server commands control operation of the computer system **2** on which the network device configuration tool **10** operates as a TFTP server, i.e. is capable of sending and/or receiving TFTP messages via the bootp/TFTP interface **30**. Finally, the view network activity log provides a historical display of exchanges between the network device configuration tool **10** and network devices requesting configuration.

Commands under the window menu are "arrange", "configuration files" "workspace", "requesting router" and "network devices". The arrange command is a pull-down menu which provides a set of commands which modify the appearance of the configuration management GUI **100**. The configuration files command displays the configuration files stored in the memory subsystem. The workspace and net-

work device commands respectively move the network administrator to the network workspace **102** and the device window **104**. Finally, the requesting router command provides a list of network devices **26** requesting IP addresses and configuration files from the network device configuration tool **10**.

The command button bar **110** provides immediate execution of selected commands available from the pull-down menus **108**. The commands which may be executed from the command button bar **110** are new, open, save, print, draw mode, move mode, network devices, workspace, requesting router, view network activity log and help.

Proceeding to step **52**, the network administrator executes the command selected at step **50**. For example, if the network administrator decides to retrieve an existing network configuration map stored in memory, the network administrator may click on the "open map" command button on the command button bar to display a list of map files **16** stored in memory and then select a map file to be opened. An exemplary network configuration map **106** which may be stored in memory is illustrated in FIG. **4**. The network configuration map **106** is comprised of a vendor specific device **112**, here, a modular router manufactured by Compaq Computer Corporation of Houston, Tex., having a first peripheral connection interface (or "PCI") slot coupled to a first ethernet-type LAN **114**, a second PCI slot coupled to a second ethernet-type LAN **116**, a third PCI slot coupled to a frame relay-type WAN **118** and an ethernet port coupled to a third ethernet-type LAN **120**.

Continuing on to step **54**, the network administrator then decides whether to edit the network configuration map **106** displayed in the network workspace **102**. If the network administrator decides not to edit the network configuration map **106**, the method proceeds to step **56** where the network administrator decides whether to execute another command. If so, the method returns to step **56**. Otherwise, the network administrator closes the network configuration tool at step **58** to end the method.

Returning now to step **54**, if the network administrator decides to go to the network workspace **102** to edit either the blank map initially loaded into the network workspace **102** at step **48** or, if a saved map was retrieved from the map files **16** by executing an "open file" command at step **52**, the retrieved map loaded into the network workspace at step **52**, the method proceeds to step **59** (FIG. 3B) where the network administrator decides whether to edit the map displayed in the network workspace **102**. If the network administrator decides not to edit the map, the method returns to step **56** (FIG. 3A). If, however, the network administrator decides to edit the configuration network map **106** displayed in the network workspace **102** the method proceeds to step **60** where editing of the map commences.

At step **60**, the network administrator may select a device type displayed in device type window **104** and add a device of the selected type to the map **106** displayed in network workspace **102**. Proceeding to step **62**, to add a device of a type displayed in the device type window **104** to the network configuration map **106** displayed in the network workspace **102**, the user selects an icon representing a desired device type and, using the "drag and drop" process, places the icon on the network configuration map **106** displayed in the network workspace **102**. For example, using a mouse or other conventional pointing device, the user would point to an icon representing the desired device type, select the device type by holding a leftmost button on the mouse in the depressed position, point to the desired position on the map and release the button. By doing so, a new device of the selected type is added to the network map. For example, in FIG. **7**, a single network device, i.e., a modular router **122** manufacture by Compaq Computer Corporation, and a pair of network entities, i.e., ethernet type LANs **124** and **126** have been added to the network configuration map **106**.

Each network device and/or network entity added to the network configuration map **106** is associated with a corresponding one of the configuration scripts **12**-N. Accordingly, at step **64**, the map editor **14** displays the name of the network device or entity contained in the attributes section **34** of the corresponding configuration script **12**-N as the name of the newly added network device or entity. For example, the name of the network device **122** added to the network configuration map **106** is "Compaq Router".

Upon placing the, as yet unconnected, network device **122** and entities **124**, **126** on the network configuration map **106**, or if it was decided at step **60** to not add a network device or entity to the network configuration map **106**, the method proceeds to step **66** where the network administrator decides whether to connect the newly added network devices and entities **122**, **124** and **126** to other network devices or entities. For example, the network administrator may decide to connect the Compaq router **122** to the frame relay-type WAN **118**, the ethernet-type LAN **124** and the ethernet-type LAN **126**. If the network administrator decides to connect the Compaq router **122** to the ethernet-type LAN **124**, the method proceeds to step **68** where the network administrator would select the Compaq router **122** by holding a leftmost button on the mouse in the depressed position while pointing to the Compaq router **122**, draw a connection between the Compaq router **122** and the ethernet-type LAN **124** by repositioning the mouse to point at the ethernet-type LAN **124** while the button is depressed and releasing the button to complete the connection.

Continuing on to step **70**, the map editor **14** determines whether the proposed connection is permissible. If the proposed connection is permitted, the line drawn by the network administrator is completed at step **72**. The connection interface(s) for the origination device are then placed on the network configuration map **106** and the method continues on to step **74** for further editing of the network configuration map **106**. For example, as shown in FIG. **7**, PCI slot **1** of the Compaq router **122** has been used to connect the device to the ethernet-type LAN **126**, PCI slot **2** to connect to the frame relay-type WAN **118** and PCI slot **4** to connect to the ethernet-type LAN **124**. If, however, the proposed connection is not permitted, the line drawn by the user is deleted at step **76** before continuing on to step **74**.

Returning to step **70**, the method by which the map editor **14** determines whether the proposed connection is permitted will now be described in greater detail. An initial determination as to whether the proposed connection is permissible is made based upon the contents of the attributes section **34** of the configuration scripts **12**-N for the devices placed on the map **106**. For example, the configuration script for a Cisco 2514 router is set forth in the attached appendix. A portion of the attributes section of the configuration script contains the following code:

```
CONNECT  "ETHERNET0"  "Ethernet"
CONNECT  "ETHERNET1"  "Ethernet"
CONNECT  "SERIAL0"    "X.25" "Frame Relay" "PPP" "HDLC"
CONNECT  "SERIAL1"    "X.25" "Frame Relay" "PPP" "HDLC"
```


This portion of the configuration script code contains considerable connection information for the device.

Specifically, the device has four connection interfaces—two ethernet ports and two serial ports. Furthermore, the two ethernet ports are only connectable to an ethernet-type LAN entities device while the two serial ports are connectable only to X.25, frame relay, PPP and HDLC entities. Accordingly, at step **70**, the mapper compares the list of network device or entity types which are connectable for the two devices and/or entities for which connection is proposed. If the devices and/or entities are connectable, the method proceeds to step **72** where connection of the two devices and/or continues.

Turning momentarily to FIG. **3D**, the step of connecting the two devices and/or entities will now be described in greater detail. The method commences at step **150** and continues on to step **152** where the configuration file for the origination device or entity is reviewed to determine if the origination device or entity has an available slot which is connectable to the destination device or entity and to step **154** where the configuration file for the destination device or entity is reviewed to determine if the destination device or entity has an available slot which is connectable to the origination device or entity. If either the origination or destination device or entity do not have an available slot which is connectable to the other device or entity, a determination is made at step **156** that the devices/entities cannot be connected. The proposed connection is then deleted at step **158** and, continuing on to step **166**, the method returns to step **72**.

Returning to step **154**, if it is determined that both the origination and destination devices or entities have available slots, the method proceeds to step **160** where a connection interface is selected for the originating device and on to step **162** where a connection interface is selected for the destination device or entity. At both of these steps, the network administrator may select any one of a list of available connection interfaces overlayed on the network configuration map **106** by the network device configuration tool **10**. If only one connection interface is available for a device or entity, however, the map will automatically select the available interface and indicate its selection of the connection interface to the network administrator. Upon selecting connection interfaces for the devices or entities, the method proceeds to step **164** where the selected connection interface for the device **122** is displayed on the network configuration map **106** and on to step **166** for a return to step **72**.

Upon either a decision not to connect devices or entities at step **66**, a completion of a proposed connection at step **72** or a deletion of a proposed connection at step **76**, the method proceeds to step **74** where the network administrator decides whether to configure a device. To initiate configuration of a selected unconfigured device, the network administrator double clicks on the device to be configured. At step **78** (FIG. **3C**) the configuration guide **18** retrieves the guided configuration section **40** from the configuration script **12**-N for the type of device to be configured and, proceeding to step **80**, executes the script commands contained in the general script commands portion **40***a* of the guided configuration section **40**. In turn, the execution of the script commands causes a series of questions to be asked of the network administrator, the answers to which are used to construct a configuration file. For example, if the script commands set forth in the guided configuration section of the configuration script set forth in Appendix A were executed during configuration of a Cisco 2514 router, the network administrator would be asked to name the router, indicate whether to configure internet protocol (or "IP") for the router, indicate which IP routing protocol should be used for the router, whether to configure IPX for the router, indicate whether the router should be password protected, choose a password for the router, indicate whether the configuration mode for the router should be password protected and choose a password for the configuration mode.

Proceeding to step **82**, the configuration guide **18** determines whether any ports of the device being configured are connected to a second device or entity. If any of the ports are connected, the method proceeds to step **84** where the configuration guide **18** executes the script commands for the connected ports. For example, if serial port 1 of a Cisco router 2514 was connected to a WAN, the configuration guide **18** would execute the script commands set forth in serial1 portion of the script commands set forth in Appendix A. Thus, in this example, the network administrator would be asked whether the serial port should be configured, the IP address and mask for the port, the IPX network number, whether the port should be configured for frame relay, the type of connector being used for the port, the local data link connection identifier (or "DLCI"), the Committed Information Rate (or "CIR") and the Excess Information Rate (or "EIR") for the port and whether to use compression.

The configuration guide **18** collects the information necessary to configure the device by engaging the network administrator in a dialog during which the configuration guide **18** generates a series of GUIs, each of which displays a request for information and provides areas in which the requested information may be inputted and buttons for guiding the network administrator through the dialogue. By way of example, an IP address GUI **200** is illustrated in FIG. **5**. The network administrator may input the IP address and mask for the indicated slot and device by respectively entering the IP address and mask in areas **202** and **204**. The network administrator may also review a prior GUI in the dialogue by depressing button **206**, proceed to the next GUI in the dialogue by depressing button **208**, request help by depressing button **210** or exit the configuration dialog by depressing button **212**.

Upon successful execution of the script commands for the connected ports at step **84**, or if it was determined at step **82** that no ports are connected for the device being configured, the configuration dialog is completed at step **86** and, at step **88**, the information provided by the network administrator during the dialogue is used to construct a local configuration file **20** for the device. If desired, the network administrator may view the local configuration file **20** constructed during this process at step **90**, directly edit any of the configuration commands contained therein at step **92** before saving the constructed local configuration file **20** to the memory subsystem and associating it with the device. Selected portions of the configuration information contained in the local configuration file **20** may be displayed on the network configuration map **106**. For example, FIG. **7** displays the IP address and mask for PCI slot **1** of the Compaq router **122** which was input by the network administrator during configuration of the device. The network configuration map **106** may also include a indicator **128**, for example, a loop surrounding a device, which indicates that a device has been configured.

Having successfully constructed a local configuration file **20** and associated it with the device being configured, the method proceeds to step **96** (FIG. **3B**) where the network administrator decides whether to upload the configuration file to the device. If upload is selected, the method proceeds to step **97** where the constructed configuration file is uploaded to the network device **26**. Various mechanisms may be used to upload a constructed configuration file to the

network device **26**. For example, in many circumstances, an in-band transfer of the configuration file via telnet may be used. In other circumstances, other mechanisms more fully described below may be necessary to transfer configuration information to the network device **26**.

While constructing a local configuration file for a device, the network device configuration tool **10** also constructs a bootptab file for the device. The bootptab file is particularly useful in those situations where the network administrator decides not to upload the configuration file upon completing the construction thereof, for example, if the network device is unconnected, powered down or otherwise unavailable. A bootptab file for a device contains the serial number for the device to be configured, an IP address to assign to the device to be configured and the configuration file to be uploaded to the device. As will be more fully described with respect to FIGS. **8–9**, below, the bootptab file provides information necessary for unattended remote configuration of network devices as they are connected to the network.

Returning now to FIG. **3B**, after completing upload of the configuration file at step **97**, or if the network administrator decided at step **96** not to upload the configuration file, the method proceeds to step **98** where the network administrator decides whether to perform subsequent configuration on a device on the network configuration map **106**. If subsequent configuration of a device is selected, the method proceeds to step **99** where subsequent configuration of a selected device is performed from a backplane bitmap of the selected device. To select a device for subsequent configuration, the network administrator double clicks on a configured device included on the network configuration map **106**. By doing so, a bitmap of the backplane of the selected configured device is displayed.

FIG. **6** illustrates a backplane bitmap **220** for the Compaq router **122** of FIG. **7**. As may now be seen, the various connection interfaces used to connect the router **122** to network entities, as well as unconnected connection interfaces, are graphically displayed on the backplane bitmap **220** using the information contained in the bitmap section **36** of the configuration script **12-N** and the local configuration file **20** for the Compaq router **122**. Specifically, for the Compaq router **122**, PCI slot **1** has been used to provide a first ethernet connection **222**, PCI slot **2**, an HSSI connection **224** and PCI slot **4**, a second ethernet **226**. PCI slot **3**, however, remains unconnected. From the backplane bitmap **220**, the network administrator may view the settings for a port by double clicking on a selected port or, by depressing the right mouse button, bring up a pull down menu of commands contained in the network entity commands section **38***a*-x of the bitmap menu **38** for the network entity connected to the selected port and select any of the configuration commands listed on the pull down menu for execution.

After completing subsequent configuration of the device at step **99**, or if the network administrator decided at step **98** not to perform subsequent configuration, the method returns to step **56** (FIG. **3A**).

Turning next to FIG. **8**, a method of transmitting configuration information to a network device **26** in accordance with the teachings of the present invention shall now be described in greater detail. The method commences at step **250** by launching the network device configuration tool **10**. As previously stated with respect to FIG. **3A**, launch of the network device configuration tool **10** initiates the generation of the configuration manager GUI **100**. In addition, launch of the network device configuration tool **10** initiates listening, by the network device configuration tool **10** at step **252**, for the presence of unconfigured network devices **26** on the network.

Proceeding to step **254**, the network device configuration tool **10** will detect bootp packets transmitted on the network and determine if the bootp packet was issued by a device requesting configuration information from the network device configuration tool **10**. More specifically, if an unconfigured network device **26** powers up on the network, the unconfigured network device **26** will periodically issue a bootp packet which contains a medium access code (or "MAC") address for the device and a code which indicates that the device is requesting configuration information. For example, the code may be placed in the vendor specific field of the bootp packet. If a detected bootp packet does not contain a request for configuration information, the method returns to step **252** where the configuration tool continues to listen for bootp packets.

If, however, the network device configuration tool **10** determines at step **256** that the issuing device is requesting configuration information, for example, by matching a request code held by the network device configuration tool **10** with a corresponding code contained in the detected bootp packet, the method proceeds to step **258** where the network device configuration tool **10** will determine if the device requesting configuration information has a corresponding bootptab file **32** and if the description of the device requesting configuration information matches the device drawn on the network configuration map **106**.

In order to determine whether the device requesting configuration information has a corresponding bootptab file **32** and if the description of the device matches the device drawn on the network configuration map, the attributes section **34** must be modified to include two additional portions—bootpid and subdeviceid. The bootpid portion contains a number unique to a particular device type and model number. The subdeviceid identifies the type of devices installed in the device. For example, if the network device was a modular router having 4 PCI slots, each connectable to ethernet, X.25, frame relay, PPP and IDSN type entities, and an ethernet port connectable to an ethernet entity with a ThunderLan board connectable to ethernet entities, a W-Adapter connectable to X.25, frame relay and PPP entities and a Basic Rate ISDN Board connectable to ISDN entities installed therein, the attributes section **34** could be as set forth in the following code:

```
:ATTRIBUTES
ICON ROUTER.ICO
NETENTITY "Compaq Router"
DESCRIPTION "Modular and Fast"
CONNECT   "PCI SLOT 1"  "Ethernet" "X.25"  "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "PCI SLOT 2"  "Ethernet" "X.25"  "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "PCI SLOT 3"  "Ethernet" "X.25"  "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "PCI SLOT 4"  "Ethernet" "X.25"  "Frame Relay"
                        "PPP" "ISDN"
CONNECT   "Ethernet Port" "Ethernet"
bootpid 103
subdeviceid 11 "ThunderLan board" "Ethernet"
subdeviceid 12 "W-Adapter" "X.25" *2 "Frame Relay"
          *2 "PPP" *2
subdeviceid 13 "Basic Rate ISDN Board" "ISDN"
© 1995 Compaq Computer Corporation
```

The guided configuration section would be similarly modified to include an additional command script portion which, upon execution thereof, will issue any additional requests for information, for example, installed devices, necessary to construct the bootptab file described herein such that a determination as to whether the description of the device

requesting configuration matches the device drawn on the network configuration map **106**.

Proceeding to step **260**, if the device requesting configuration has a matching bootptab file, i.e., the bootptab file has a bootpid which matches the serial number of a device having a bootptab file and if the devices installed in the device requesting configuration match the devices identified in the subdeviceid portion of the configuration file for the matching bootptab file, the network device configuration tool **10** issues a bootp reply at step **260**. The bootp reply contains the filename which matches the configuration file described in the matching bootptab file. Using the filename contained in the bootp reply, at step **262**, the device requesting configuration may issue a TFTP request for configuration information to the network device configuration tool **10** which identifies the configuration file containing its configuration information.

Continuing on to step **264**, in response to the TFTP request containing the filename of a configuration file issued by the device requesting configuration, the network device configuration tool **10** responds by issuing the requested configuration file to the device. At step **266**, the unconfigured network device configures itself using the information contained in the configuration file transmitted thereto by the network device configuration tool **10** and, at step **268**, the method ends.

Returning to step **258**, if the device requesting configuration does not have a matching bootptab file, the method proceeds to step **270** where the network device configuration tool **10** generates a pop-up requesting device GUI **300** which overlays a portion of the configuration manager GUI **100**. A requesting device GUI **300** is illustrated in FIG. 9. As illustrated herein, the requesting device GUI **300** includes an icon representing the unconfigured network device **302** requesting configuration.

Proceeding to step **272**, the network administrator may select one of two options to configure the device requesting configuration. If the network administrator decides that the device **302** is a new device, the requesting device may be dropped onto the network workspace **102**, thereby adding the requesting device to the network configuration map **106** as an unconnected device. Proceeding on to step **274**, the method would return to step **64** (FIG. 3B) wherein the previously discussed process of constructing a configuration file and uploading the configuration file to the unconfigured network device may be completed to configure the device requesting configuration.

Returning to step **272** and, now proceeding to step **276**, the network administrator may instead opt to drop the device **302** requesting configuration onto an existing device, for example, router **112**, already included on the network configuration map **106**. By dropping the device **302** requesting configuration onto an existing device on the network configuration map **106**, the network administrator is indicating that the device **302** requesting configuration is the same device that is already on the network configuration map **106** but, due to a difference between the description of the device **302** in the bootptab and the description of the device **112** contained in the corresponding configuration file, the network device configuration tool **10** is unable to recognize that the two are the same device.

Proceeding on to step **278**, the network device configuration tool **10** would reconcile the configuration file and the bootptab file for the device. If the two are irreconcilable, the method terminates at step **280**. If the two can be reconciled, the configuration file is revised appropriately at step **282** and the method then returns to step **264** so that the network device configuration tool **10** may issue the revised configuration file to the device **302** requesting configuration in the manner previously described. To reconcile the device **302** requesting configuration and an existing device such as the router **112**, the network device configuration tool **10** reviews the devices installed on the device requesting configuration match the devices installed. If the installed devices match, then the configuration file is modified using the contents of the bootptab file. The method then proceeds to step **264** so that the network device configuration tool **10** may issue the revised configuration file to the device **302** requesting configuration.

Thus, there has been described and illustrated herein a configuration manager for network devices and an associated method of providing configuration information to a network device. However, those skilled in the art will recognize that many modifications and variations besides those specifically mentioned may be made in the techniques described herein without departing substantially from the concept of the present invention. Accordingly, it should be clearly understood that the form of the invention described herein is exemplary only and is not intended as a limitation on the scope of the invention.

-A-1- Patent Application
Docket # CMPQ-0986
P-986

APPENDIX "A"

```
File: 2514.DEV

:ATTRIBUTES
ICON ROUTER.ICO
NETENTITY "Cisco 2514"
DESCRIPTION "Cisco IOS"

CONNECT "ETHERNETO" "Ethernet"
CONNECT "ETHERNET1" "Ethernet"
CONNECT "SERIAL0"   "X.25" "Frame Relay" "PPP"  "HDLC"
CONNECT "SERIAL1"   "X.25" "Frame Relay" "PPP"  "HDLC"

:GUIDED_CONFIG

     use 2514vars.use
     frame
     askstring routername "What do you want to name this
router?" minlen 2 maxlen 20
     savefilename $routername "-confg"
     addconfig "hostname  " $routername

     frame
     radio configip "Do you want to configure IP on this
router?" "Yes" "No"
     if $configip = "Yes" then
                    frame
                      radio routeprot "What IP routing
protocol do you want to use?"
"RIP"  "IGRP"
                    addconfig " !"
                    addconfig " ! IP routing protocol to
use"
                    addconfig " !"
                    addconfig "router  " $routeprot
                    addconfig " !"
     endif

     frame
     radio configipx "Do you want to configure IPX on
this router?" "Yes" "No"
     if $configipix = "Yes" then
                    addconfig " !"
                    addconfig " ! Enable IPX routing"
                    addconfig " !"
```

-A-2- Patent Application
Docket # CMPQ-0986
P-986

```
                    addconfig "ipx routing"
                    addconfig " !"
                    frame
                    radio yesno "Do you want password
protect configuration mode?"
"Yes" "No"
                    if $yesno = "Yes" then
                              askpass enablepassword
"Enter password for
configuration mode" minlen 5 maxlen 20
                              addconfig " !"
                                addconfig  "enable
password " $password
                              addconfig " !"
                    endif

     endif
     frame
     addconfig " !"
     addconfig "no ip domain-lookup"
     addconfig " !"

;ETHERNET0

     frame
     assign portname "Ethernet0"
     use 2514eth.use

;ETHERNET1

     frame
     assign portname "Ethernet1"
     use 2514eth.use

;SERIAL0

     frame
     assign portname "SERIAL0"
     use 2514wan.use

;SERIAL1

     frame
     assign portname "SERIAL1"
     use 2514wan.use
```

```
;BITMAP
bitmap          2514.bmp

locate              "ETHERNET0"    56    69    87    79
locate              "ETHERNET1"    110   69    141   79
locate              "SERIAL0"      250   64    302   82
locate              "SERIAL1"      326   64    377   82

:BITMAP_MENU

;Ethernet

menu "Something"
     menuitem "Pick me"
          define string 80
          askstring string "Enter something"

          addconfig "you entered " $string
menuend

;Serial0

menu "No items available yet"

;PROMPTS
"Password;"
"-More-"
```

File: 2514vars.use

```
# variables to use with the Cisco 2514 config scripts
(2514*.*)
define configip          3
define configipx         3
define routeprot         4
define password          30
define enablepassword    30
define ipaddress         15
define ipmask            15
define ipxaddress        20
define ipxencap          15
define lpsz              80
define portname          10
define yesno             3
define frconnector       10
```

-A-4- Patent Application
Docket # CMPQ-0986
P-986

```
define frdici          3
define frcir           4
define freir           4
define frportspeed     4
define pppmtu          5
define pppauth         4
define pppcompress     10
define ppplapb         3
define ppptacacs       3
define pppconnector    5
define routername      30
```

File: 2514eth.use

```
     assign lpsz "Do you want to configure " $portname
"?"
     radio yesno $lpsz "Yes" "No"
     if $yesno = "Yes" then

          addconfig  "  !"  $Sportname  "configuration
commands"
          addconfig " !"
          addconfig "interface " $portname
          addconfig " !"

          if $configip = "Yes" then

               frame
               getip ipaddress ipmask
               askip ipaddress ipmask "Enter IP network
that interface is plugged
 into"

               addconfig " !"
               addconfig " ! IP related commands"
               addconfig " !"
               addconfig "ip address " $ipaddress " "
$ipmask
               addconfig " !"
               assignip $ipaddress $ipmask

          endif

          if $configipx = "Yes" then
```

```
               frame
               getipx ipxaddress
               askstring ipxaddress "Enter IPX network
number for this interface"
hex maxlen 8
               frame
                 radio  ipxencap  "What  type  of  IPX
ethernet encapsulation should be
used?"  "ARPA" "Novell-Ether"  "SAP"  "SNAP"
               addconfig " !"
               addconfig " ! IPX related commands"
               addconfig " !"
                addconfig "ipx network " $ipxaddress
               addconfig "ipx encap " $ipxencap
               addconfig " !"
               assignipx $ipxaddress

          endif
     endif
```

File: 2514wan.use

```
     assign lpsz "Do you want to configure " $portname
"?"
     radio yesno $lpsz "Yes" "No"

     if $yesno = "Yes" then

          addconfig " !"
          addconfig  "  !  "   $portname  "  configuration
commands"
          addconfig "interface " $portname
          addconfig " !"

          if $configip = "Yes" then

          frame
          assign lpsz "Enter IP address for " $portname
          askip ipaddress ipmask $lpsz
          addconfig "!"
          addconfig "! IP related commands"
          addconfig "!"
          addconfig "ip address " $ipaddress " " $ipmask
     endif

     if $configipx = "Yes" then
          frame
```

-A-6- Patent Application
Docket # CMPQ-0986
P-986

```
        assign lpsz "Enter IPX network number for "
$portname
        askstring ipxaddress $lpsz hex maxlen 8
        addconfig " !"
        addconfig " ! IPX related commands"
        addconfig " !"
        addconfig "novell address " $ipxaddress
        addconfig " !"
    endif

    if $connected = "Frame Relay" then

        frame
        assign lpsz "Would you like to configure Frame
Relay for "
$portname "?"
        assign yesno "Yes"
        radio yesno $lpsz "Yes" "No"
        if $yesno - "Yes" then

            addconfig " !"
            radio yesno $lpsz "Yes" "No"
            if $yesno = "Yes" then

                addconfig " !"
                addconfig " ! Set Encapsulation to
Frame Relay"
                addconfig "encapsulation frame-relay"

#               radio frconnector "What connector
type are you using?" "RS-232" "V.35"
                askstring frdlci "What is your local
DLCI" MIN 16 MAX 996
                # set DLCI command
                assign lpsz "What is " $portname
"Physical Port Line Speed (Kbps)"
                askstring frportspeed $lpsz MIN 1.2
MAX 2048
                # set port speed command
                assign lpsz "What is the Committed
Information Rate (CIR)
for " $portname "?"
                askstring frcir $lpsz MIN 1.2 MAX
2048
                # set CIR command
```

```
                    assign lpsz "What is the Excess
Information Rate (EIR) for *
$portname "?"
                    askstring freir $lpsz MIN 1.2 MAX
2048
                    # formula goes here
                    # set EIR command
                    radio yesno "Do you want to use
compression?" "Yes" "No"
                    # if goes here
                    # set compression command
#OTHER PARAMETERS to set (some under advanced button)
#frportspeed
#frcir
#frier
#frremname
#frconnrtr
#frcompress
#frnettype
#frmaxframe
#frcongmonper
#frmeasint
#frlit
#frpvt
#frfsef
#fret
#frcet
#frclocking
#fremulate
#fremulate
#frlinestate
#map
                    display "<more frame relay questions
would normally follow>"
               endif
          else
          if $connected = "PPP" then

               frame
               assign lpsz "Would you like to configure
PPP for " Sportname "?"
               assign yesno "Yes"
               radio yesno $lpsz "Yes" "No"
               if $yesno = "Yes" then

# Set defaults
```

-A-8- Patent Application
Docket # CMPQ-0986
P-986

```
                    assign ppmtu "1500"
                    assign pppauth "CHAP"
                    assign pppcompress "Stacker"
                    assign ppplapb "No"
                    assign ppptacacs "No"

                    addconfig " !"
                    addconfig " ! Set Encapsulation to
PPP"
                    addconfig "encapsulation ppp"

#               radio pppconnector "What connector type
are you using?"  "RS-232"
"V.35"

                    frame
                    radio pppauth "What authentication
protocol are you using?"
"PAP"  "CHAP"  "Off"
                    if $pppauth <> "Off" then
                         addconfig " !"
                         addconfig " ! PPP Authorization
related
commands"
                         addconfig " !"
                         addconfig "no ppp
authentication"
                    endif
               endif

               frame
               radio pppcompress "What compression method
are you using?"
"Predictor" "Stacker" "Off"
               if $pppcompress <> "Off" then
                    addconfig " !"
                    addconfig " ! PPP Compression related
commands"
                    addconfig " !"
                    addconfig "ppp compress "
$pppcompress
               else
                    if $pppcompress = "Off" then
                         addconfig " !"
                         addconfig " ! PPP Compression
related commands"
```

```
                    addconfig " !"
                    addconfig "no ppp compression"
               endif
          endif

          frame
          radio ppplapb "Use LAPB with PPP for
reliable link?" "Yes"
"No"
          if $ppplapb = "Yes" then
               addconfig " !"
               addconfig " ! PPP LAPB with PPP for
reliable link related commands"
               addconfig " !"
               addconfig "ppp reliable-link"
          else
               addconfig " !"
               addconfig " ! PPP LAPB with PPP for
reliable link
related commands"
               addconfig " !"
               addconfig "no ppp reliable-link"
          endif
          frame

          radio ppptacacs "Use TACACS to verify PPP
authentication?"
"Yes" "No"
          if $ppptacacs = "Yes" then
               addconfig " !
               addconfig " ! Use TACACS to verify
PPP authentication
               addconfig "ppp use-tacacs"
          else
               addconfig " !"
               addconfig " ! Use TACACS to verify
PPP authentication"
               addconfig " !"
               addconfig "no ppp use-tacacs"
          endif

               frame
               assign lpsz "What is the Maximum
Transmission Unit for "
$portname "?"
```

-A-10- Patent Application
Docket # CMPQ-0986
P-986

```
                askstring pppmtu $lpsz MIN 64 MAX
4096
                addconfig " !
                addconfig   "   !   Set   Maximum
Transmission Unit"
                addconfig "mtu " $pppmtu
                addconfig "ip mtu " $pppmtu
            endif

        else

            display $portname " is connected to a "
$connected " network"
            display "appropriate guided configuration
commands would appear here"
        endif
    endif
```

What is claimed is:

1. For a computer system having a processor subsystem and a memory subsystem coupled by a system bus for bi-directional exchanges therebetween, a configuration manager for configuring a network device remotely coupled thereto, said configuration manager comprising:

at least one configuration script stored in said memory subsystem, said configuration script containing a series of executable instructions for constructing a configuration file and a bootptab file for a first specified type network device, said configuration script including at least a first section containing a series of configuration commands which generate requests for information, and a second section containing a set of connection rules for connecting said first specified type of network device to at least one other specified type of network device, and where said second section of said configuration script includes at least (i) an identifier for each connection interface of said first specified type of network device and (ii) a list of network device types that can be connected to the connection interface associated therewith, said list being provided for each of said identifiers;

a first software module, executable by said processor subsystem, for constructing a configuration file suitable for upload to a network device of said first specified type and a bootptab file suitable for identifying said network device, said first software module constructing said configuration file and said bootptab file by executing said series of executable instructions contained in said configuration script; and

a second software module, executable by said processor subsystem, for processing a configuration request issued by said network device by identifying said network device using said constructed bootptab file and configuring said network device by uploading said constructed configuration file thereto.

2. A configuration manager for configuring a network device remotely coupled thereto according to claim 1 wherein information received by said first software module in response to said requests for information is used to construct said configuration file and said bootptab file.

3. A configuration manager for configuring a network device remotely coupled thereto according to claim 1 and wherein said second section of said configuration script further comprises:

a first portion which uniquely identifies said network device; and

a second portion which identifies devices installed in said network device.

4. A configuration manager for configuring a network device remotely coupled thereto according to claim 3 and wherein said first section of said configuration script further comprises:

a first portion corresponding to each of said at least one other specified type of network device specified in said connection rules contained in said second section of said configuration script;

said first portion containing a subset of said series of configuration commands which are executed only if said network device for which said configuration file is being constructed is connected to a network device of said other specified type of network device.

5. A computer-implemented method for configuring a remotely located network device, comprising the steps of:

providing a configuration script containing a series of executable instructions for constructing a configuration file for a first specified type of network device, said configuration script including a first section containing a series of configuration commands and a second section containing a set of connection rules for connecting said first specified type of network device to at least one other specified type of network device, and where said second section of said configuration script includes at least (i) an identifier for each connection interface of said first specified type of network device and (ii) a list of network device types that can be connected to the connection interface associated therewith, said list being provided for each of said identifiers;

constructing a configuration file by executing said series of instructions contained in said configuration script;

detecting a request for configuration issued by a network device;

determining if said configuration file corresponds to said network device issuing said request for configuration;

if said configuration file corresponds to said network device, issuing a reply to said request for configuration that identifies said configuration file to said network device; and

issuing said configuration file to said network device in response to a request for configuration file which identifies said configuration file.

6. A computer-implemented method for configuring a remotely located network device according to claim 5 and further comprising the steps of:

generating requests for information by executing said series of configuration commands contained in said first section of said configuration script; and

constructing said configuration file using information received in response to said requests for information.

7. A computer-implemented method for configuring a remotely located network device according to claim 6 and further comprising the steps of:

constructing a bootptab file which contains a unique identifier and said configuration file for said network device.

8. A computer-implemented method for configuring a remotely located network device according to claim 7 wherein the step of determining if said configuration file corresponds to said network device issuing said request for configuration further comprises the steps of:

determining if said network device issuing said request for configuration has an identification code which matches an identification code contained in said bootptab file; and

determining if devices installed in said network device issuing said request for configuration matches installed devices identified in said bootptab file.

9. A computer-implemented method for configuring a remotely located network device according to claim 5 wherein the step of providing a configuration script containing a series of executable instructions further comprises the step of:

providing a configuration script which includes, within said second section, a first portion corresponding to each of said at least one other specified type of network device specified in said connection rules contained in said second section of said configuration script;

each said first portion containing a subset of said series of configuration commands.

10. A computer-implemented method for configuring a remotely located network device according to claim 9 and further comprising the step of:

executing said subset of said series of configuration commands contained in each said first portion only if said network device for which said configuration file is being constructed is connected to a network device of said other specified type of network device.

11. For a computer system having a processor subsystem and a memory subsystem coupled by a system bus for bidirectional exchanges therebetween, an apparatus for constructing configuration files for network devices, comprising:

a plurality of configuration scripts for various different types of network devices, said configuration scripts being stored in said memory subsystem, and each of the configuration scripts being used to construct a configuration file and a bootptab file, the bootptab files being constructed are used to assist with remote configuration of network devices including at least one connection statement for each connection port of the associated one of the network devices, the connection statements included in said configuration scripts comprise connection rules that specify permissible connections between the ports of the various different types of network devices and other of the network devices, the connection rules include an identifier for the associated port and a list of network devices that are permitted to connect to the associated port;

first computer program code for enabling selection of a particular one of the various different types of network devices; and

second computer program code for executing the one of said configuration scripts associated with the particular one of the various different types of network devices to produce a configuration file and a bootptab file for the particular one of the various different types of network devices.

12. An apparatus for constructing a configuration files for network devices according to claim 11, wherein the configuration scripts includes at least one connection statement for each connection port of the associated one of the network devices, and a series of commands, the series of commands associated with a particular one of said configuration scripts are executed by said second computer program when said second computer program code executes the particular one of said configuration scripts.

13. An apparatus for constructing a configuration files for network devices according to claim 12, wherein said apparatus further comprises:

third computer program code for uploading said configuration file to a network device of the particular one of the various different types of network devices.

14. An apparatus for constructing a configuration files for network devices according to claim 12, wherein the executing of the one of said configuration scripts by said second computer program code prompts a user of said apparatus to enter information, and the entered information is used by said second computer program code in producing the configuration file.

15. For a computer system having a processor subsystem and a memory subsystem coupled by a system bus for bidirectional exchanges therebetween, an apparatus for constructing configuration files for network devices, comprising:

a plurality of configuration scripts for various different types of network devices, said configuration scripts being stored in said memory subsystem, and each of the configuration scripts being used to construct a configuration file and a bootptab file, the bootptab files being constructed are used to assist with remote configuration of network devices including at least one connection statement for each connection port of the associated one of the network devices;

first computer program code for enabling selection of a particular one of the various different types of network devices;

second computer program code for executing the one of said configuration scripts associated with the particular one of the various different types of network devices to produce a configuration file and a bootptab file for the particular one of the various different types of network devices;

third computer program code for detecting a request for configuration issued by a connecting network device;

fourth computer program code for determining if a configuration file already exists for the network device based on the bootptab file associated with the connecting network device issuing the request for configuration; and

fifth computer program code for issuing a reply to the request for configuration that identifies the appropriate configuration file for the network device when said fourth computer program code determines that the configuration file already exists for the connecting network device; and

sixth computer program code for directing execution of at least a portion of said second computer code to produce a configuration file and then issuing the produced configuration file to the connecting network device when said fourth computer program code determines that the configuration file does not exist for the connecting network device.

16. An apparatus for constructing a configuration files for network devices according to claim 15, wherein the configuration scripts includes at least one connection statement for each connection port of the associated one of the network devices.

17. An apparatus for constructing a configuration files for network devices according to claim 16, wherein each of the configuration scripts further includes a series of commands, and

wherein the series of commands associated with a particular one of said configuration scripts are executed by said second computer program when said second computer program code executes the particular one of said configuration scripts.

18. An apparatus for constructing a configuration files for network devices according to claim 17,

wherein dialog screens for the user to input information are produced when said second computer program code executes the one of said configuration scripts associated with the particular one of the various different types of network devices, and the dialog screens prompt the user to enter the input information, and the input information that the user enters is used by said second computer program in producing the configuration file.

* * * * *

US005550816A

United States Patent [19]

Hardwick et al.

[11] **Patent Number: 5,550,816**

[45] **Date of Patent: Aug. 27, 1996**

[54] **METHOD AND APPARATUS FOR VIRTUAL SWITCHING**

[75] Inventors: **Ken Hardwick**, Sherwood, Oreg.; **Geoffrey C. Stone**, Minneapolis, Minn.

[73] Assignee: **Storage Technology Corporation**, Lousville, Colo.

[21] Appl. No.: **366,227**

[22] Filed: **Dec. 29, 1994**

[51] **Int. Cl.**^{6} **H04L 12/56**; G06F 13/00

[52] **U.S. Cl.** **370/60**; 370/85.13; 370/94.1; 395/650; 395/800; 395/200.02

[58] **Field of Search** 370/58.1, 58.2, 370/58.3, 60, 60.1, 61, 79, 85.13, 85.14, 94.1, 94.2, 94.3; 395/200, 325, 375, 650, 800, 500

[56] **References Cited**

U.S. PATENT DOCUMENTS

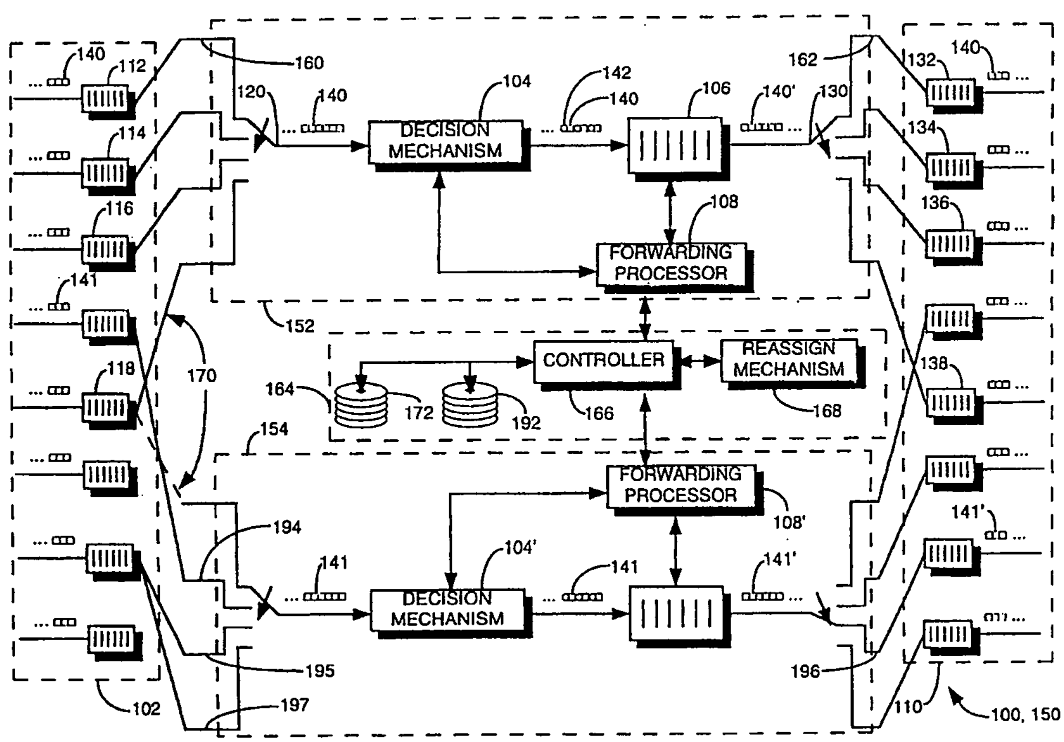
4,218,756	8/1980	Fraser	370/94.1
5,119,369	6/1992	Tanabe et al.	370/60
5,249,292	9/1993	Chiappa	395/650
5,278,834	1/1994	Mazzola	370/94.1
5,280,476	1/1994	Kojima et al.	370/60.1
5,317,568	5/1994	Bixby et al.	370/85.13
5,321,692	6/1994	Wallmeier	370/60
5,430,727	7/1995	Callon	370/85.13

Primary Examiner—Alpus H. Hsu
Attorney, Agent, or Firm—Timothy R. Schulte

[57] **ABSTRACT**

A physical switching device for use in a communication network to switch Open Systems Interconnection (OSI) network layer packets and method of use therefor is provided. The physical switching device includes at least a first and a second virtual switch. Each virtual switch includes a decision mechanism for determining an associated directive based on a destination identifier within a particular packet received at a data port. A processor is coupled to each virtual switch to insert the particular packet into an outgoing data stream on another data port to deliver the packet. Both data ports are associated with a plurality of data interfaces in the physical switching device. A management apparatus is coupled to each virtual switch to maintain information on an association between the plurality of data interfaces and the virtual switches. The management apparatus limits each processor to only inserting the particular packet on another data port associated with the same virtual switch which received the particular packet.

57 Claims, 35 Drawing Sheets



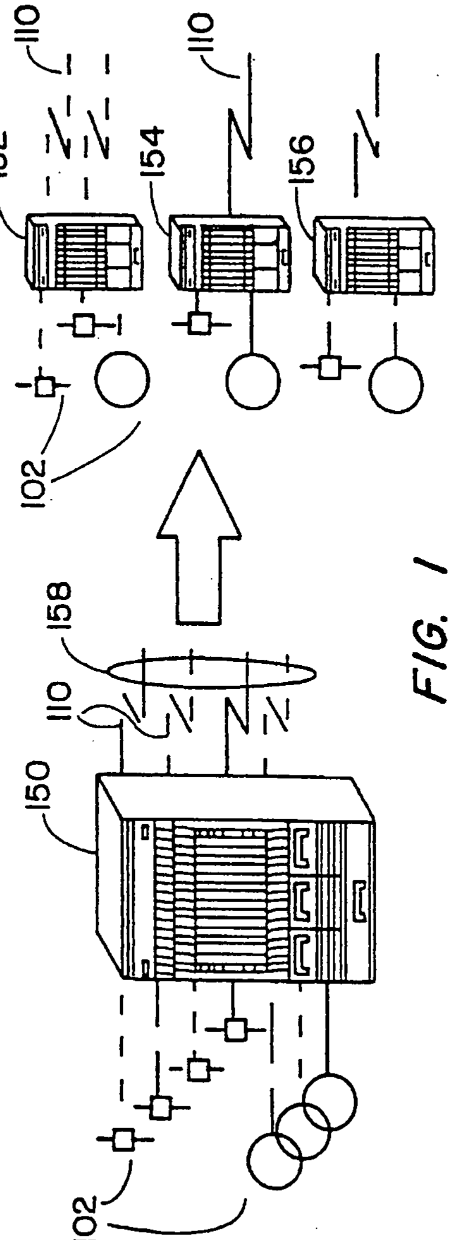

110
152
110
154
156
102
158
110
150
102

FIG. 1

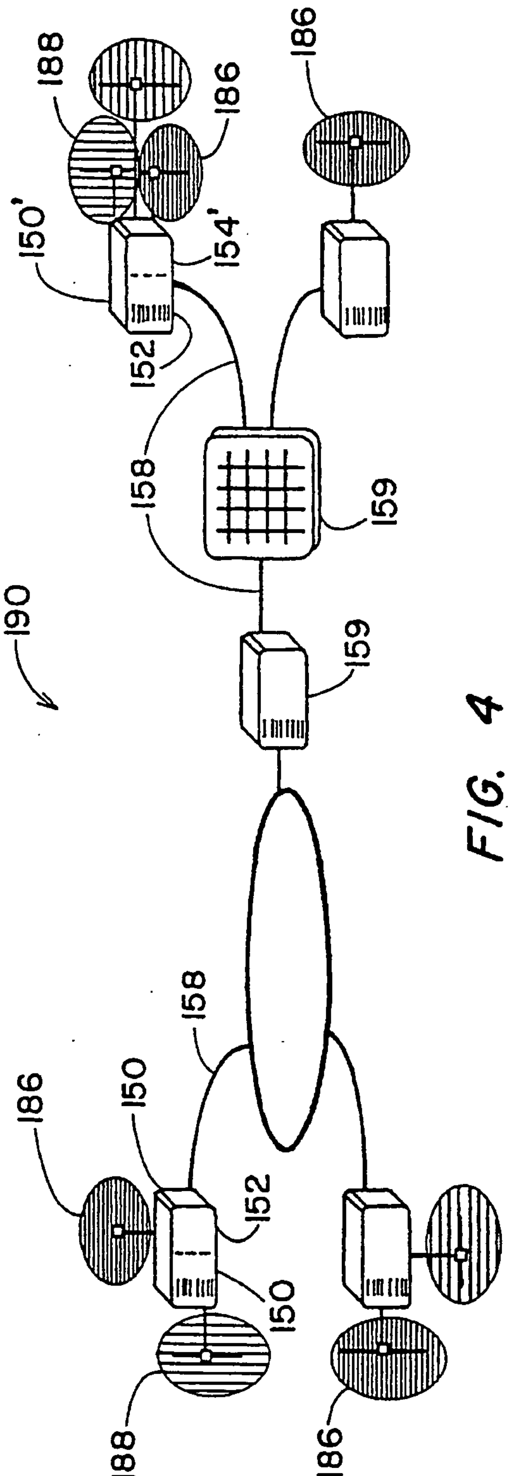

188
186
186
150'
154'
152
158
159
190
159
158
186
150
152
150
188
186

FIG. 4

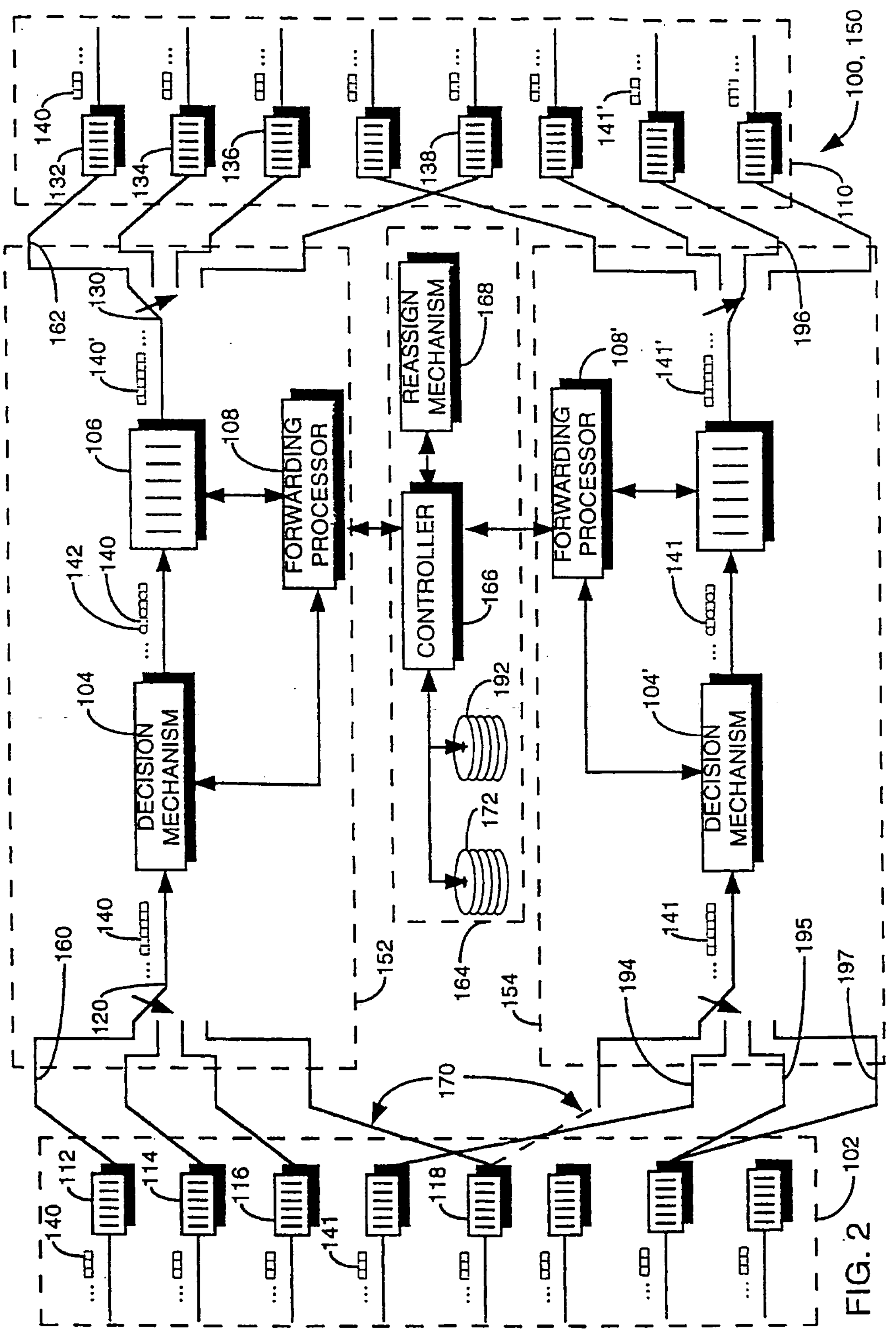

DECISION MECHANISM
FORWARDING PROCESSOR
CONTROLLER
REASSIGN MECHANISM
DECISION MECHANISM
FORWARDING PROCESSOR
100, 150
102
104
104'
106
108
108'
110
112
114
116
118
120
130
132
134
136
138
140
140'
141
141'
142
152
154
160
162
164
166
168
170
172
192
194
195
196
197

FIG. 2

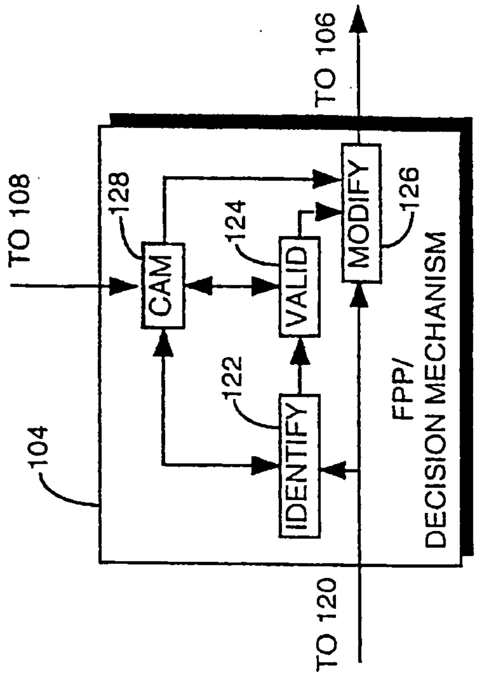

TO 108
TO 106
128
124
126
CAM
VALID
MODIFY
IDENTIFY
122
104
FPP/
DECISION MECHANISM
TO 120

FIG. 5

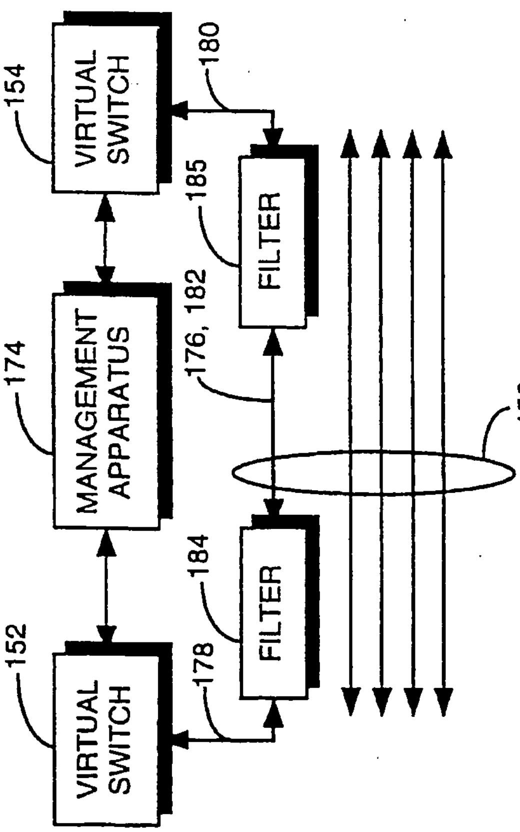

154
VIRTUAL SWITCH
180
185
FILTER
174
MANAGEMENT APPARATUS
176, 182
158
184
FILTER
152
VIRTUAL SWITCH
178

FIG. 3

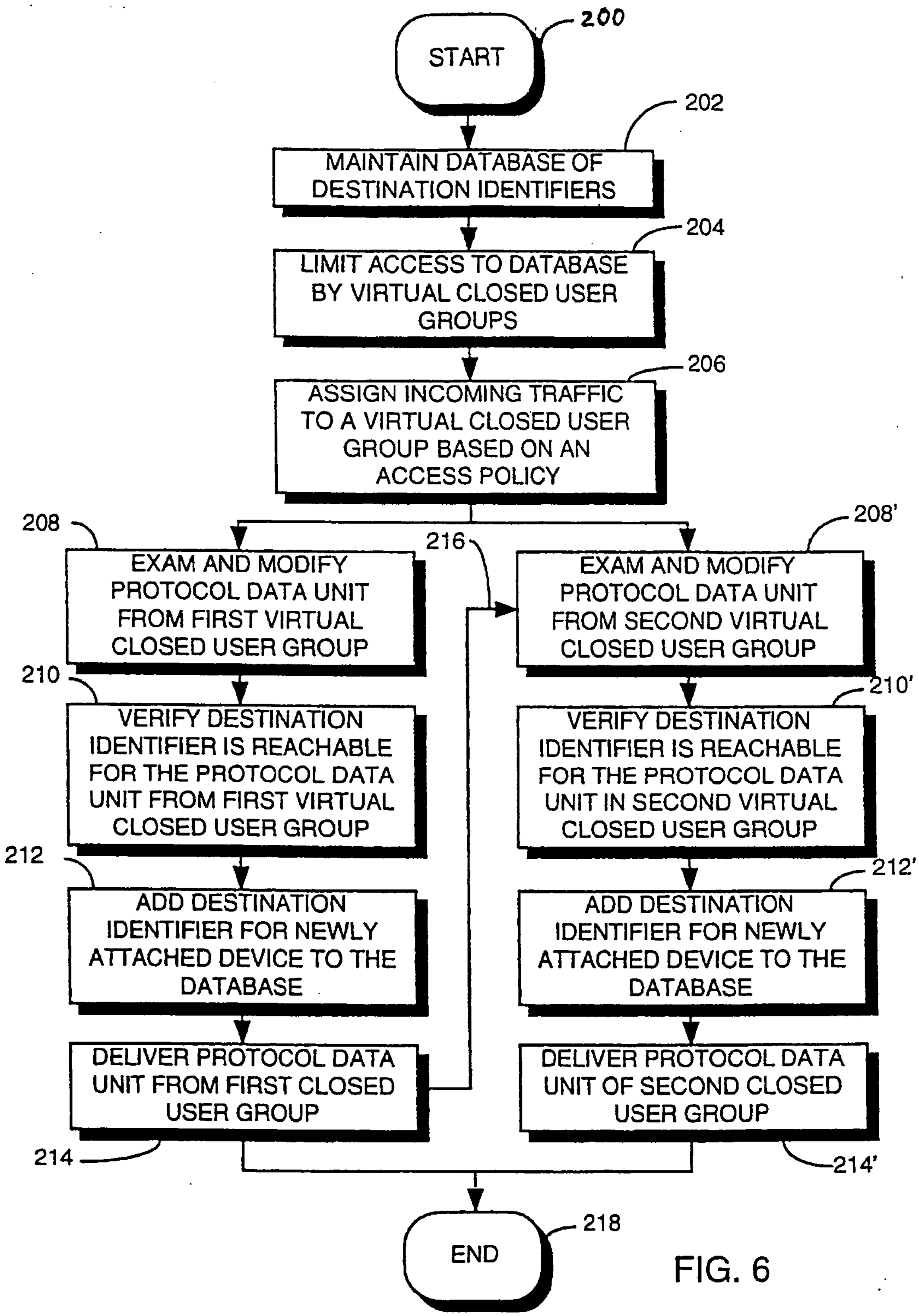

200
START
202
MAINTAIN DATABASE OF DESTINATION IDENTIFIERS
204
LIMIT ACCESS TO DATABASE BY VIRTUAL CLOSED USER GROUPS
206
ASSIGN INCOMING TRAFFIC TO A VIRTUAL CLOSED USER GROUP BASED ON AN ACCESS POLICY
208
EXAM AND MODIFY PROTOCOL DATA UNIT FROM FIRST VIRTUAL CLOSED USER GROUP
216
208'
EXAM AND MODIFY PROTOCOL DATA UNIT FROM SECOND VIRTUAL CLOSED USER GROUP
210
VERIFY DESTINATION IDENTIFIER IS REACHABLE FOR THE PROTOCOL DATA UNIT FROM FIRST VIRTUAL CLOSED USER GROUP
210'
VERIFY DESTINATION IDENTIFIER IS REACHABLE FOR THE PROTOCOL DATA UNIT IN SECOND VIRTUAL CLOSED USER GROUP
212
ADD DESTINATION IDENTIFIER FOR NEWLY ATTACHED DEVICE TO THE DATABASE
212'
ADD DESTINATION IDENTIFIER FOR NEWLY ATTACHED DEVICE TO THE DATABASE
DELIVER PROTOCOL DATA UNIT FROM FIRST CLOSED USER GROUP
DELIVER PROTOCOL DATA UNIT OF SECOND CLOSED USER GROUP
214
214'
218
END

FIG. 6

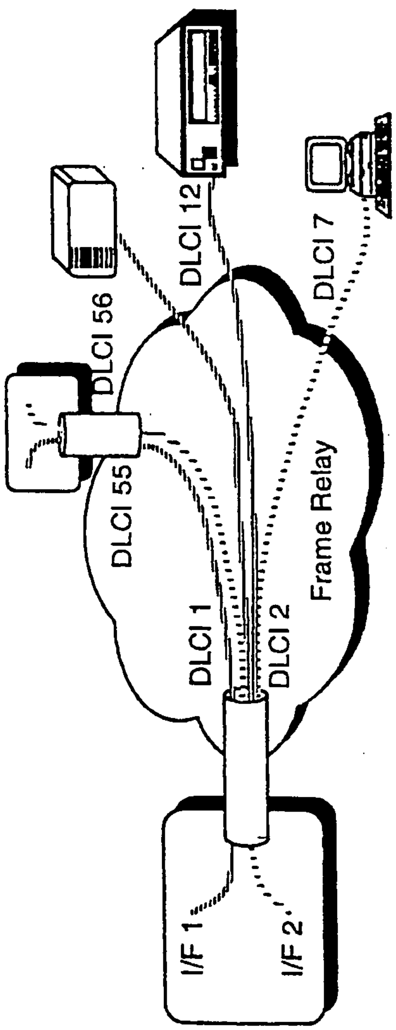

DLCI 12
DLCI 7
DLCI 56
DLCI 55
DLCI 1
DLCI 2
Frame Relay
I/F 1
I/F 2

FIG. 7

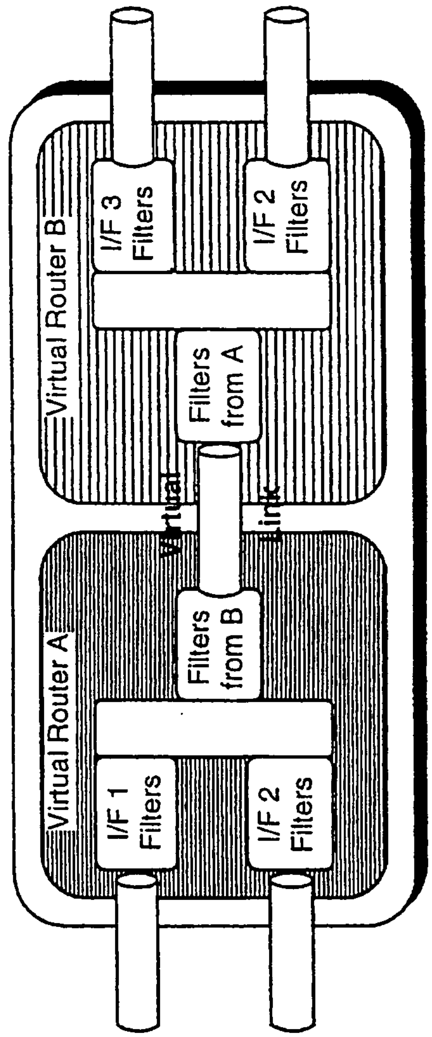

Virtual Router B
I/F 3 Filters
I/F 2 Filters
Filters from A
Virtual Link
Virtual Router A
Filters from B
I/F 1 Filters
I/F 2 Filters

FIG. 8

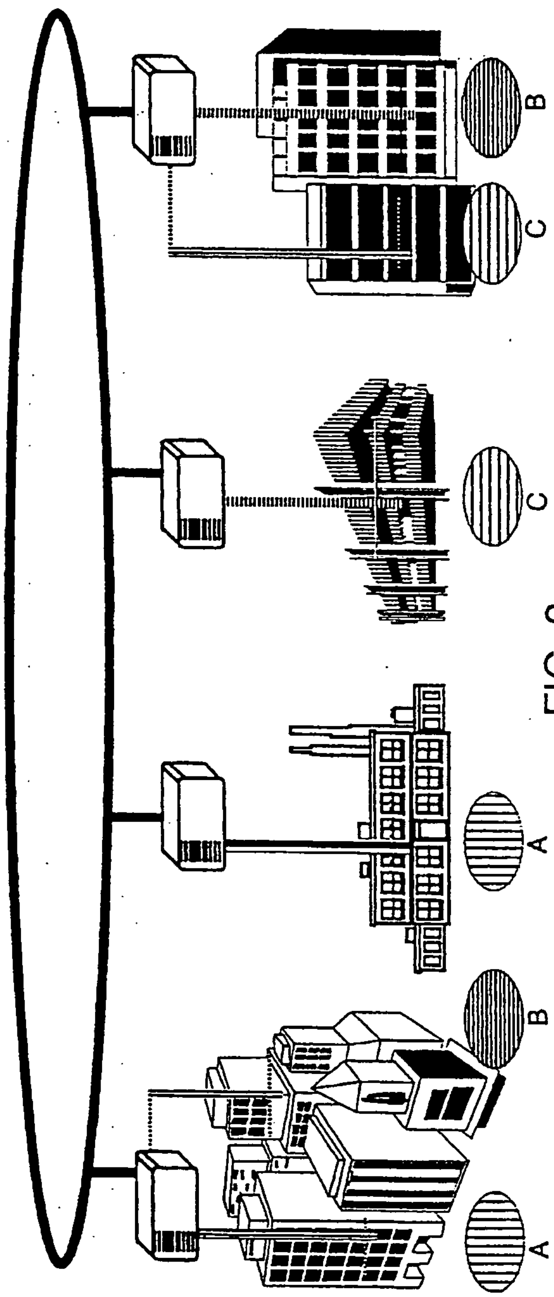


FIG. 9

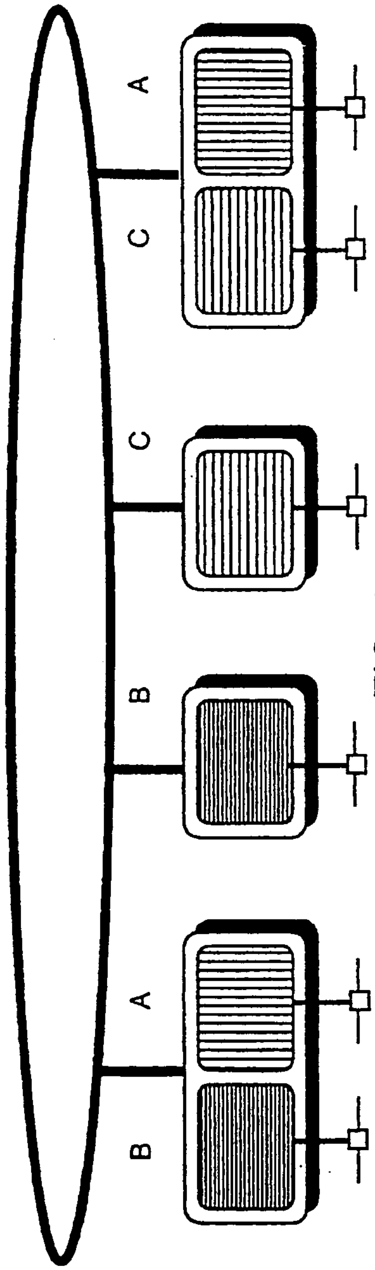


FIG. 10

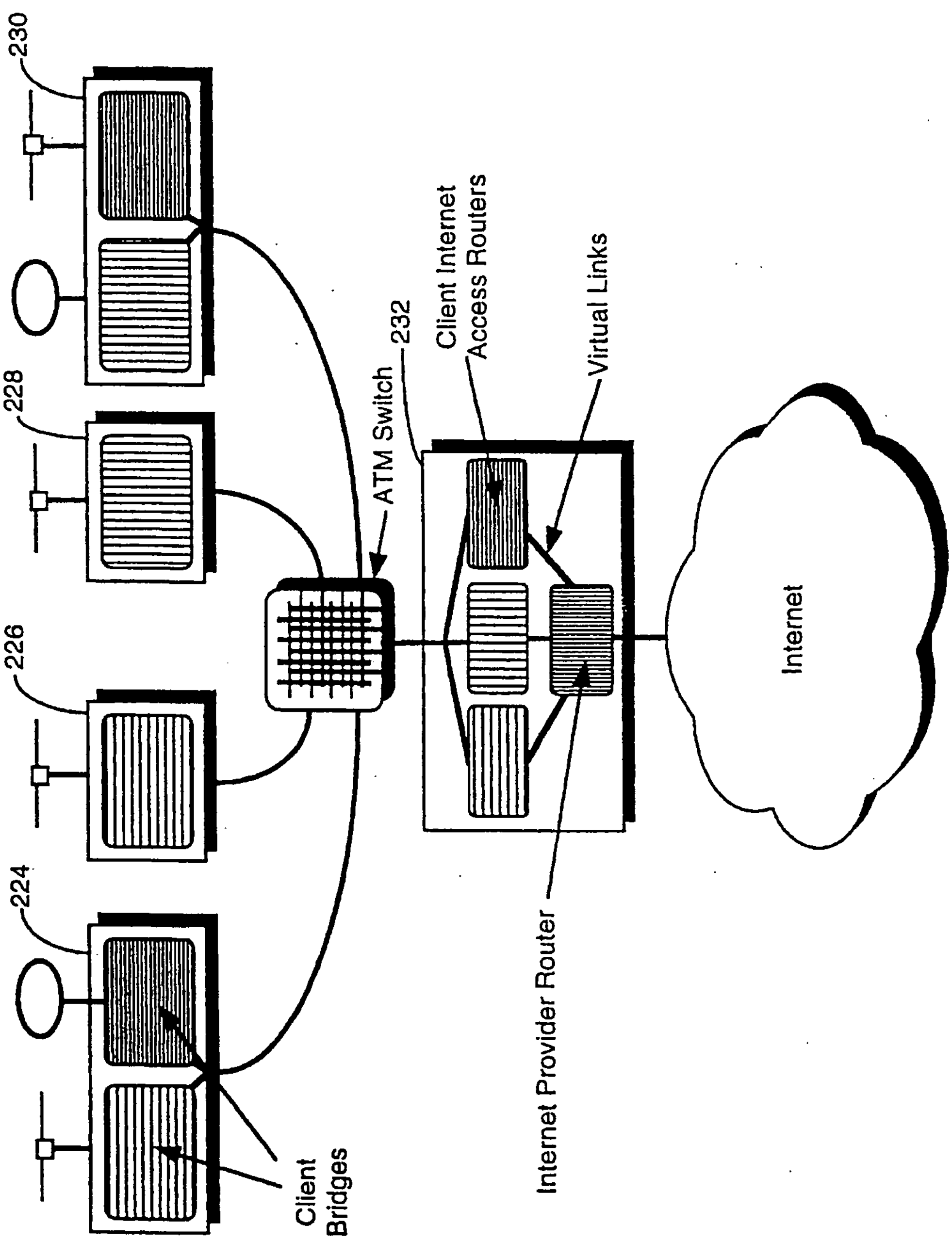

230
228
226
224
ATM Switch
232
Client Internet
Access Routers
Virtual Links
Internet
Internet Provider Router
Client
Bridges

FIG. 11

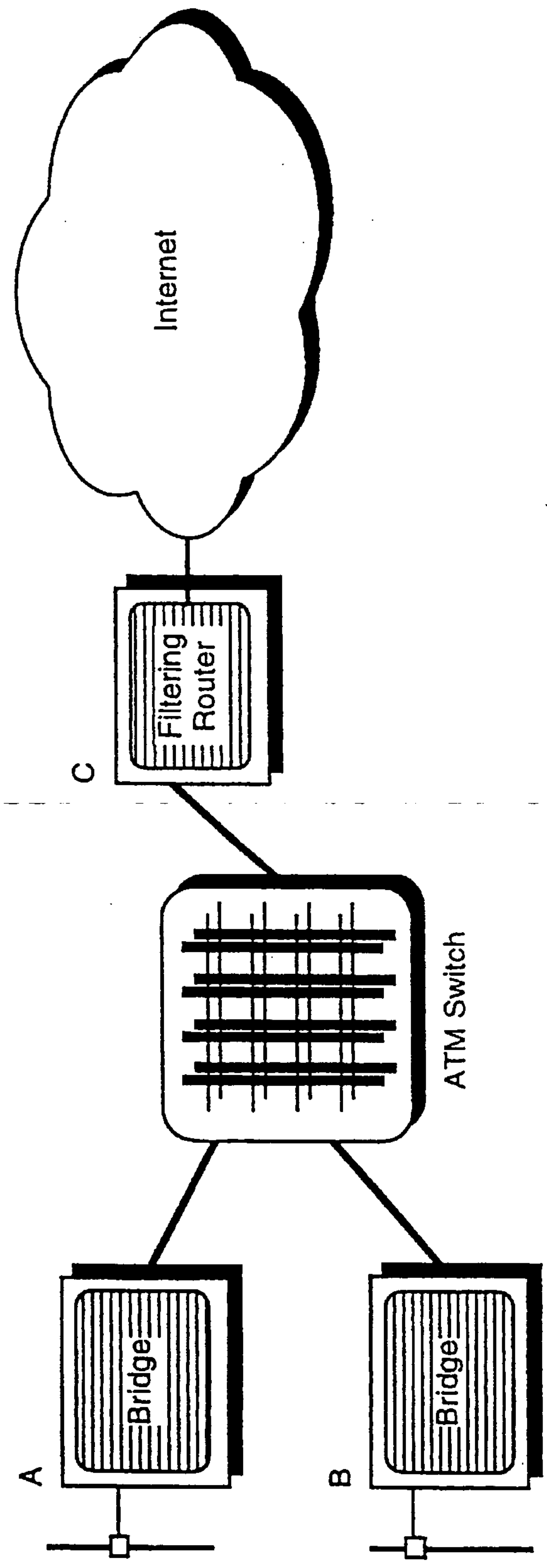


FIG. 12

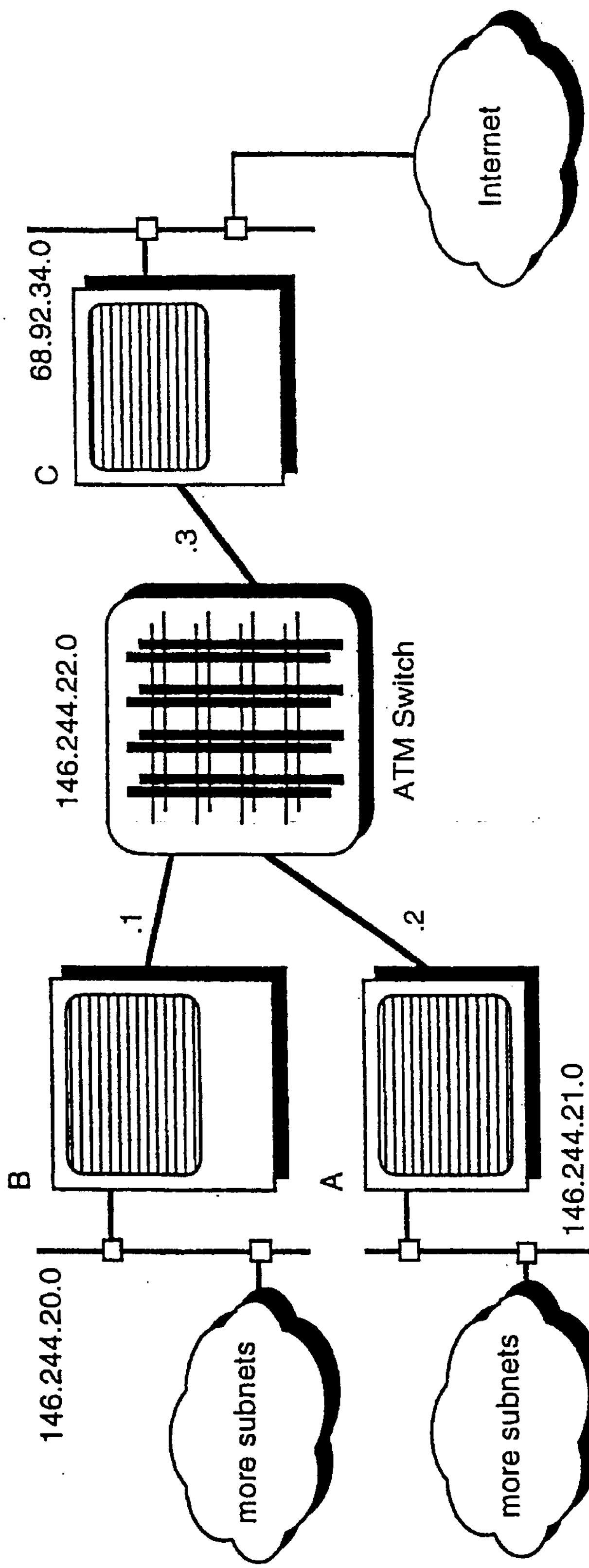

Internet
68.92.34.0
C
.3
146.244.22.0
ATM Switch
.1
.2
B
A
146.244.21.0
146.244.20.0
more subnets
more subnets

FIG. 13

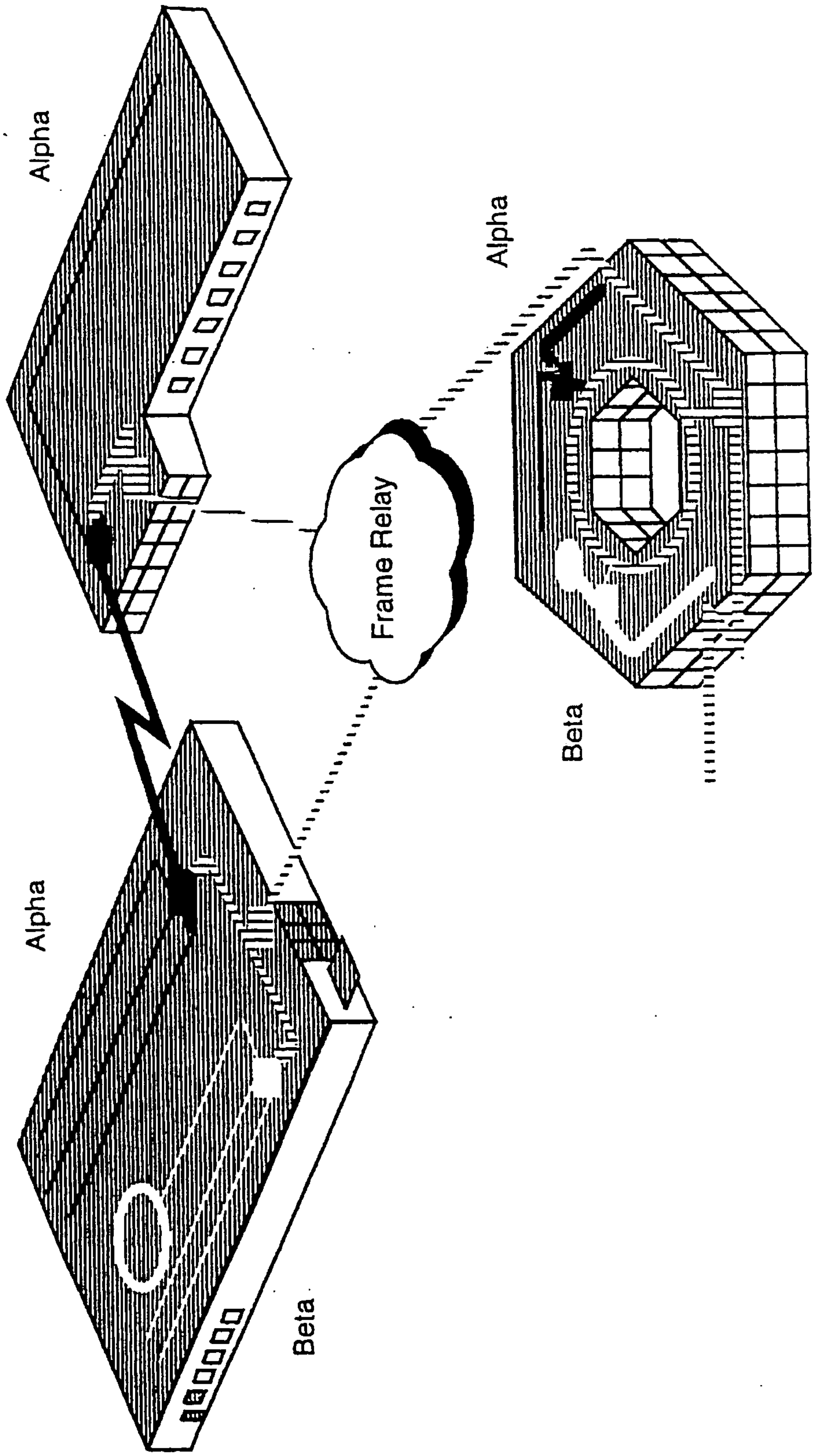


FIG. 14

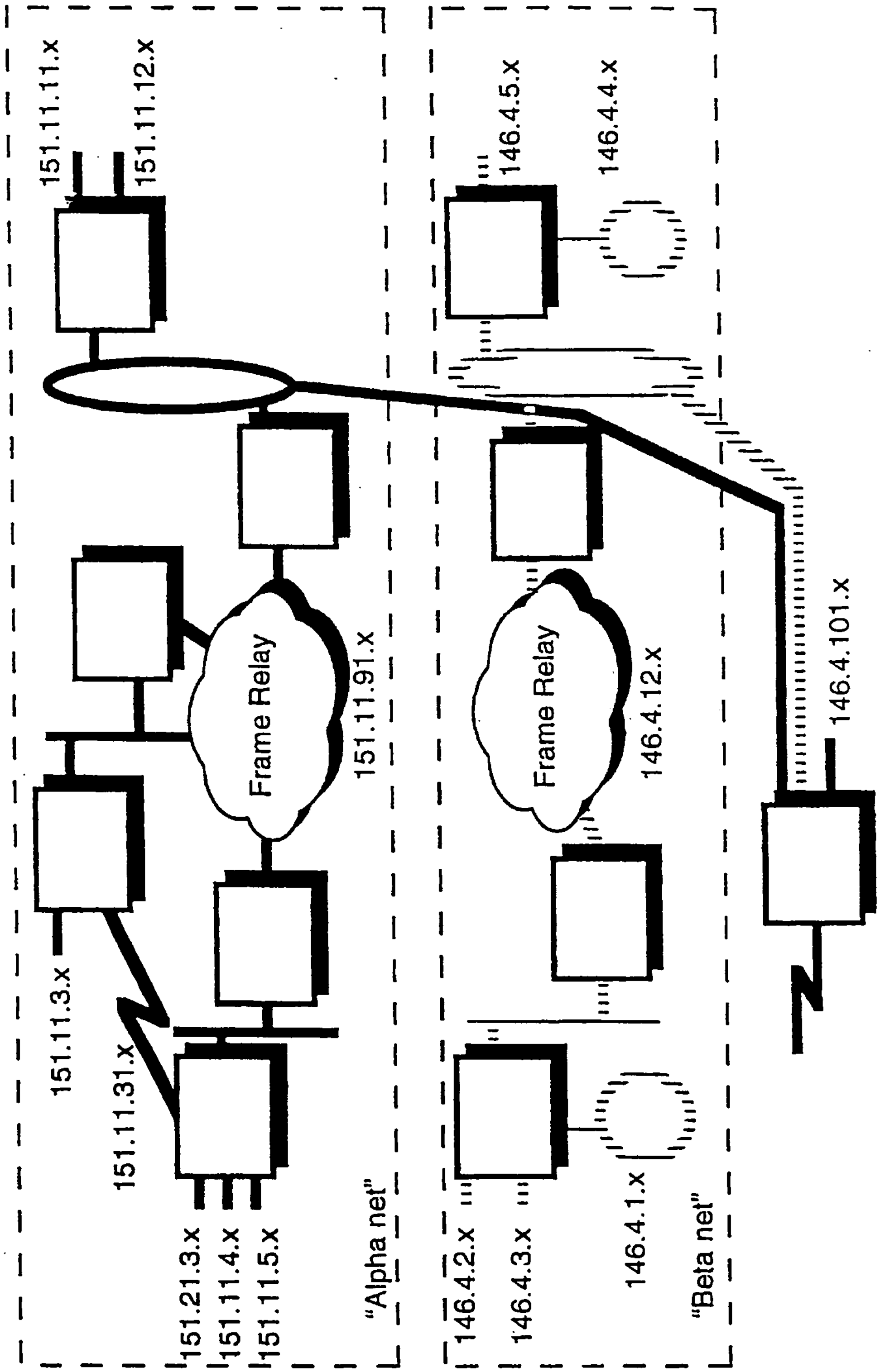

151.11.11.x
151.11.12.x
146.4.5.x
146.4.4.x
Frame Relay
151.11.91.x
Frame Relay
146.4.12.x
146.4.101.x
151.11.3.x
151.11.31.x
151.21.3.x
151.11.4.x
151.11.5.x
"Alpha net"
146.4.2.x
146.4.3.x
146.4.1.x
"Beta net"

FIG. 15

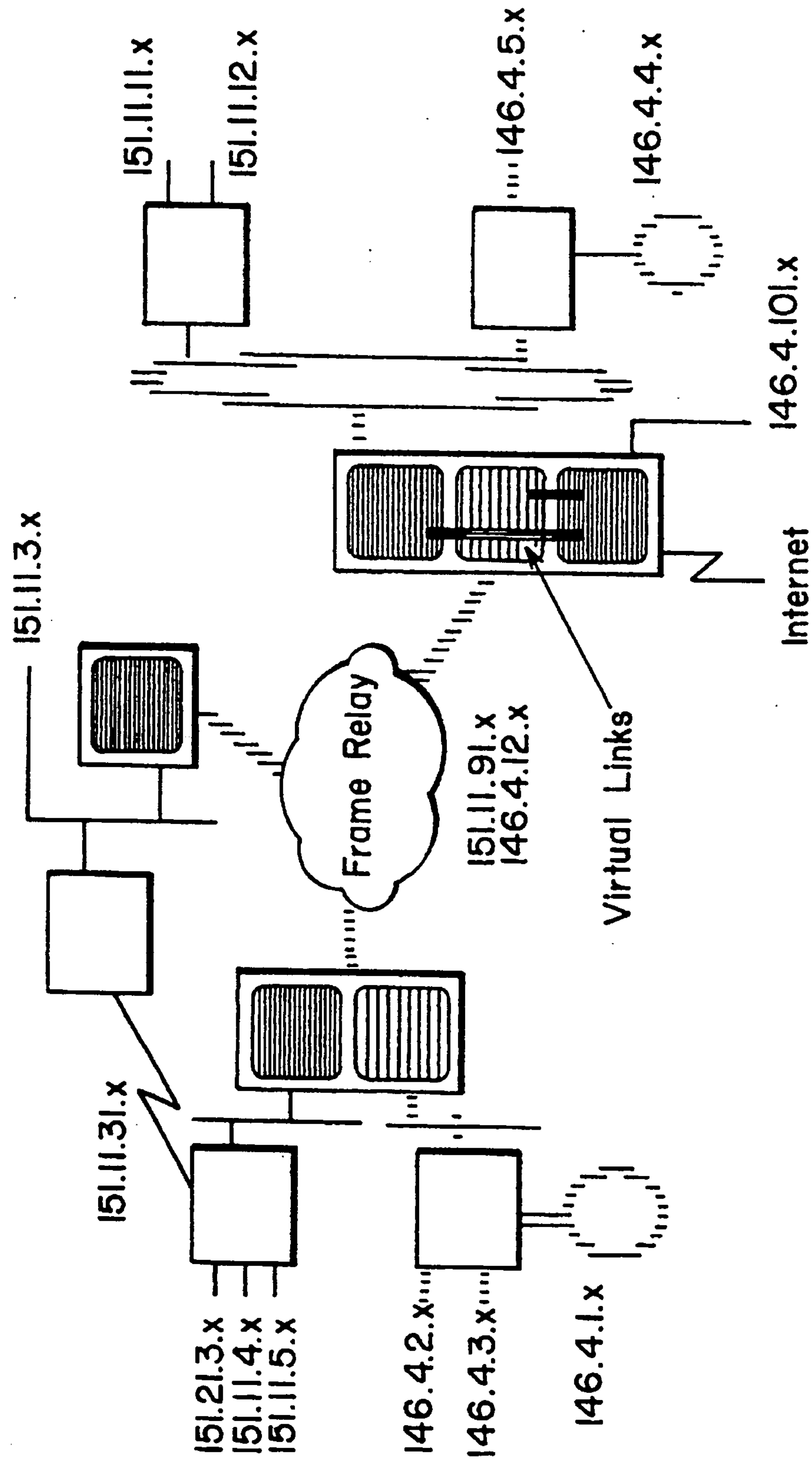

151.11.11.x
151.11.12.x
146.4.5.x
146.4.4.x
146.4.101.x
151.11.3.x
Internet
Frame Relay
151.11.91.x
146.4.12.x
Virtual Links
151.11.31.x
151.21.3.x
151.11.4.x
151.11.5.x
146.4.2.x
146.4.3.x
146.4.1.x

FIG. 16

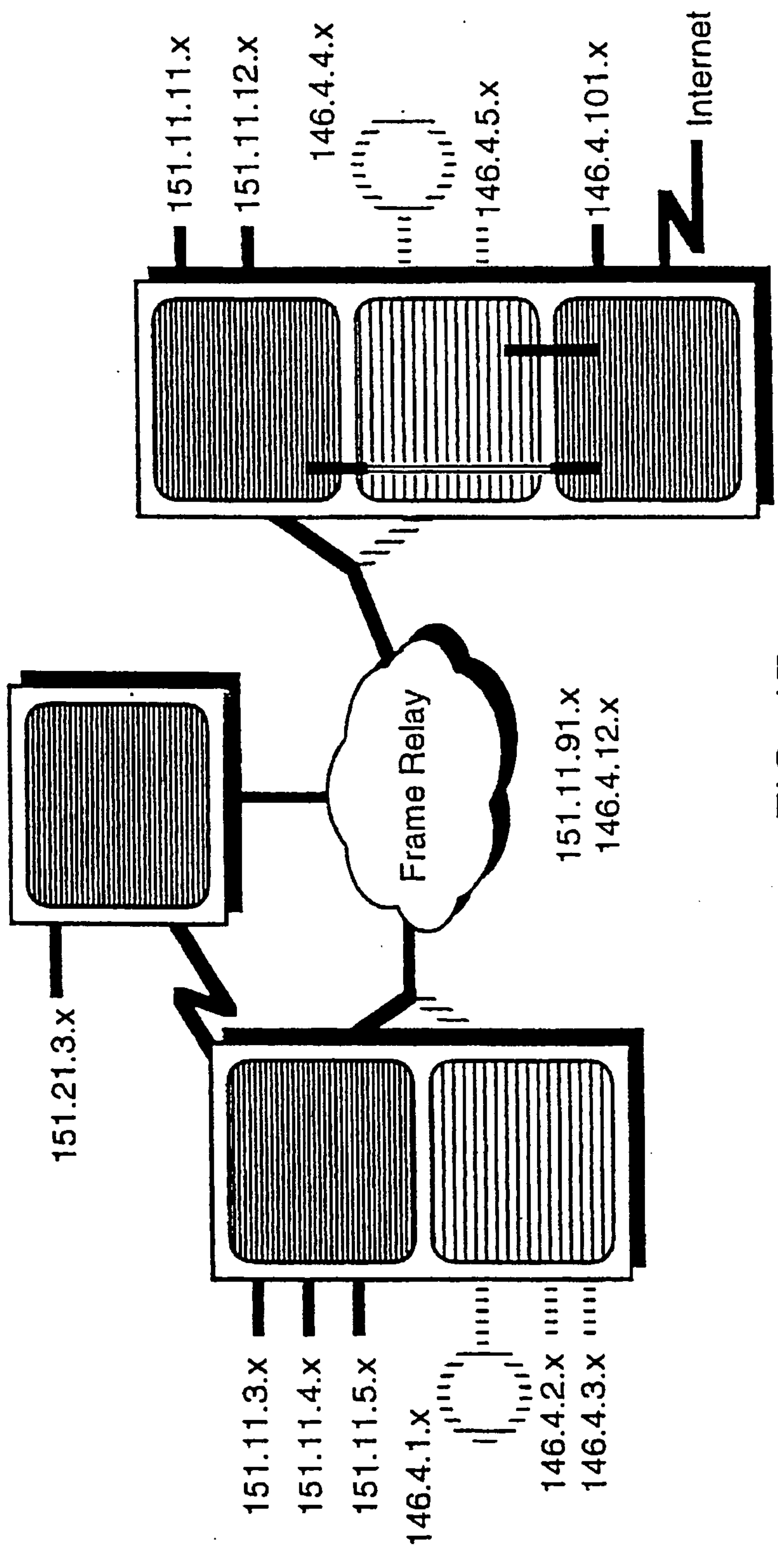

151.11.11.x
151.11.12.x
146.4.4.x
146.4.5.x
146.4.101.x
Internet
Frame Relay
151.11.91.x
146.4.12.x
151.21.3.x
151.11.3.x
151.11.4.x
151.11.5.x
146.4.1.x
146.4.2.x
146.4.3.x

FIG. 17

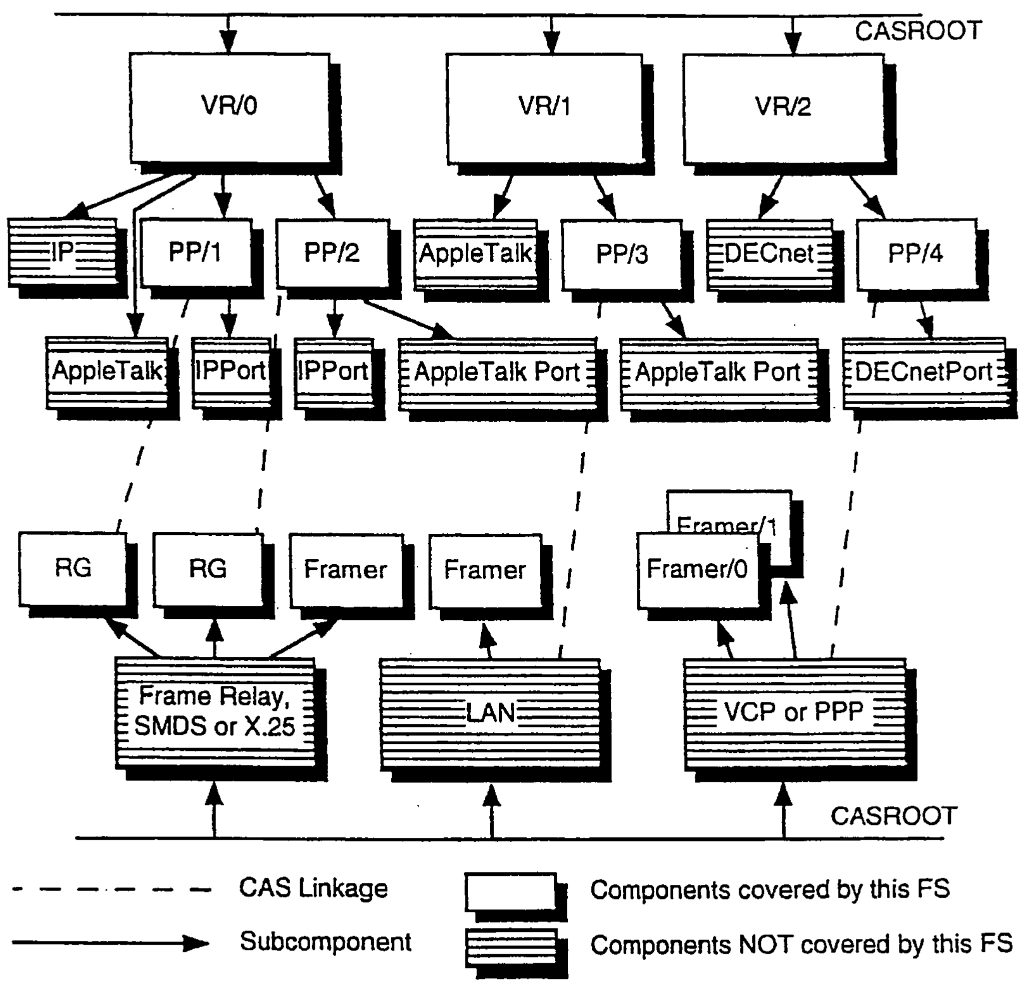

CASROOT
VR/0
VR/1
VR/2
IP
PP/1
PP/2
AppleTalk
PP/3
DECnet
PP/4
AppleTalk
IPPort
IPPort
AppleTalk Port
AppleTalk Port
DECnetPort
Framer/1
RG
RG
Framer
Framer
Framer/0
Frame Relay, SMDS or X.25
LAN
VCP or PPP
CASROOT
CAS Linkage
Subcomponent
Components covered by this FS
Components NOT covered by this FS

FIG. 18

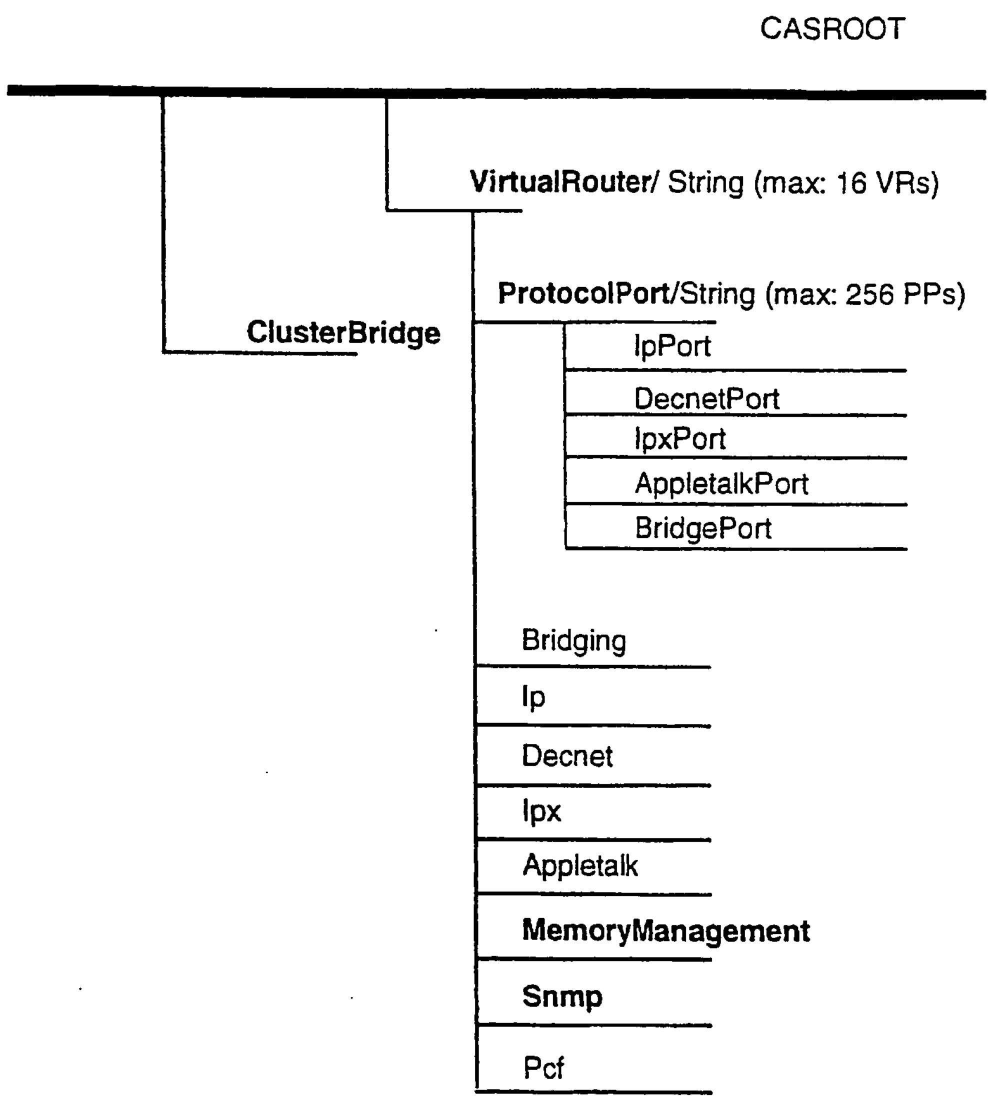

CASROOT
VirtualRouter/ String (max: 16 VRs)
ProtocolPort/String (max: 256 PPs)
ClusterBridge
IpPort
DecnetPort
IpxPort
AppletalkPort
BridgePort
Bridging
Ip
Decnet
Ipx
Appletalk
MemoryManagement
Snmp
Pcf

FIG. 19

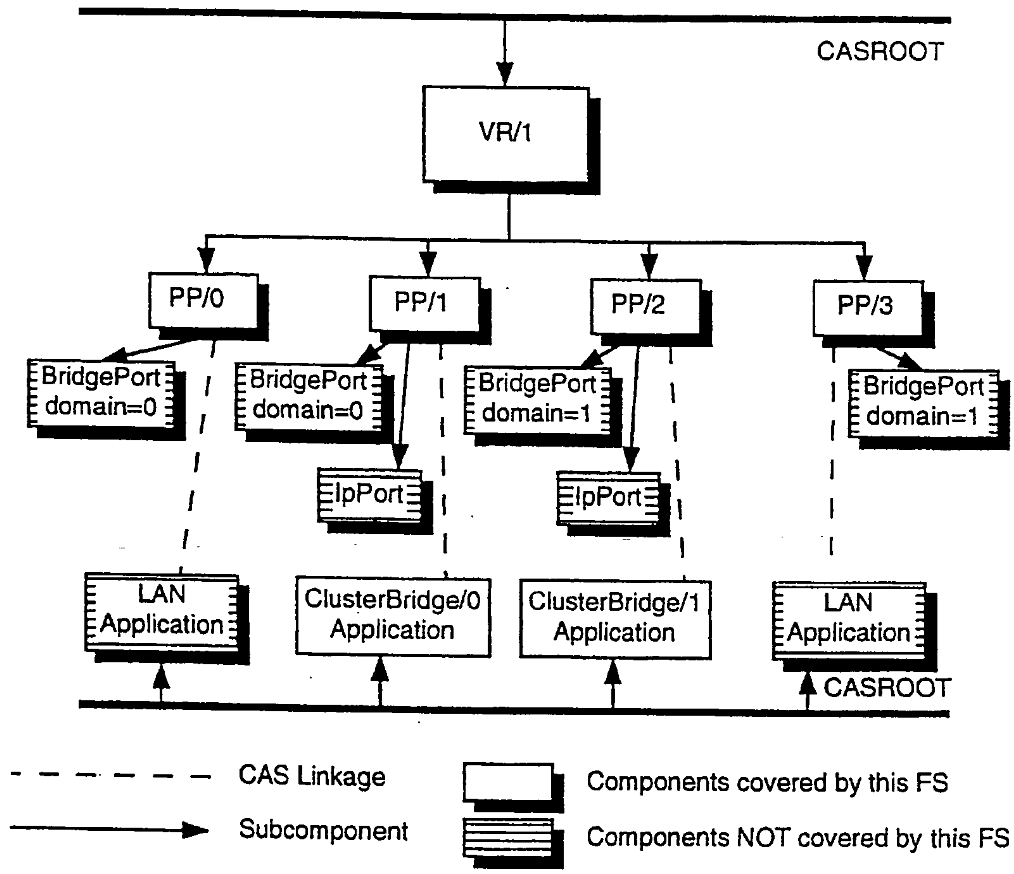

CASROOT
VR/1
PP/0
PP/1
PP/2
PP/3
BridgePort domain=0
BridgePort domain=0
BridgePort domain=1
BridgePort domain=1
IpPort
IpPort
LAN Application
ClusterBridge/0 Application
ClusterBridge/1 Application
LAN Application
CASROOT
CAS Linkage
Subcomponent
Components covered by this FS
Components NOT covered by this FS

FIG. 20

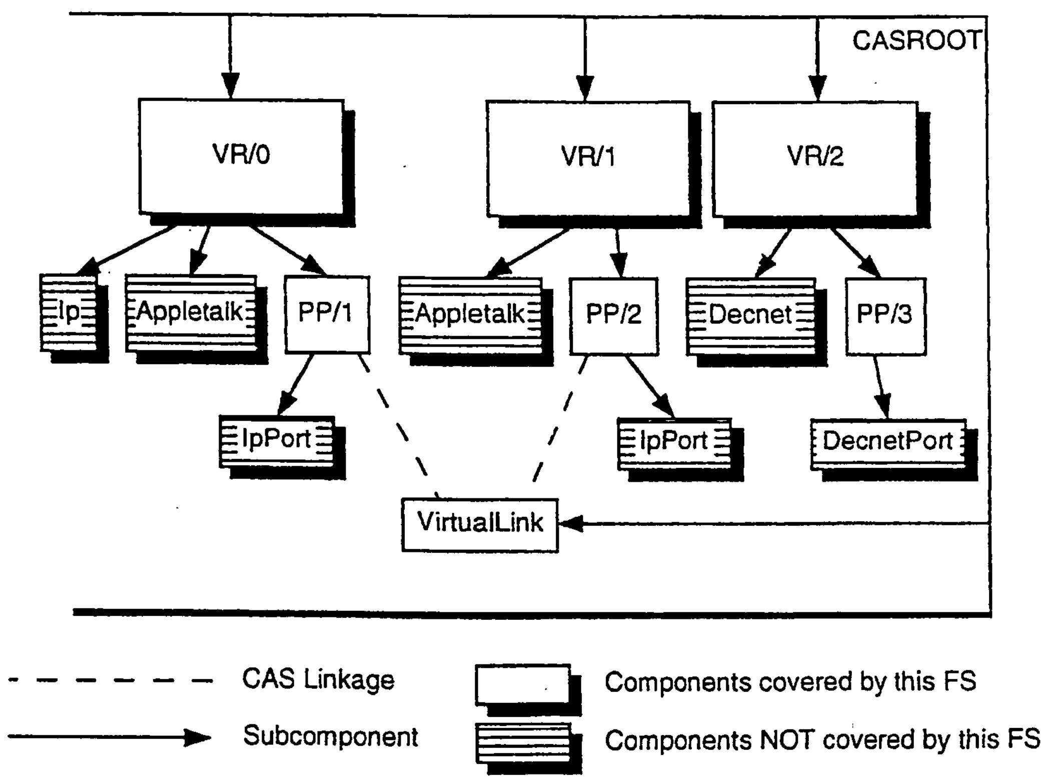

CASROOT
VR/0
VR/1
VR/2
Ip
Appletalk
PP/1
Appletalk
PP/2
Decnet
PP/3
IpPort
IpPort
DecnetPort
VirtualLink
CAS Linkage
Subcomponent
Components covered by this FS
Components NOT covered by this FS

FIG. 21

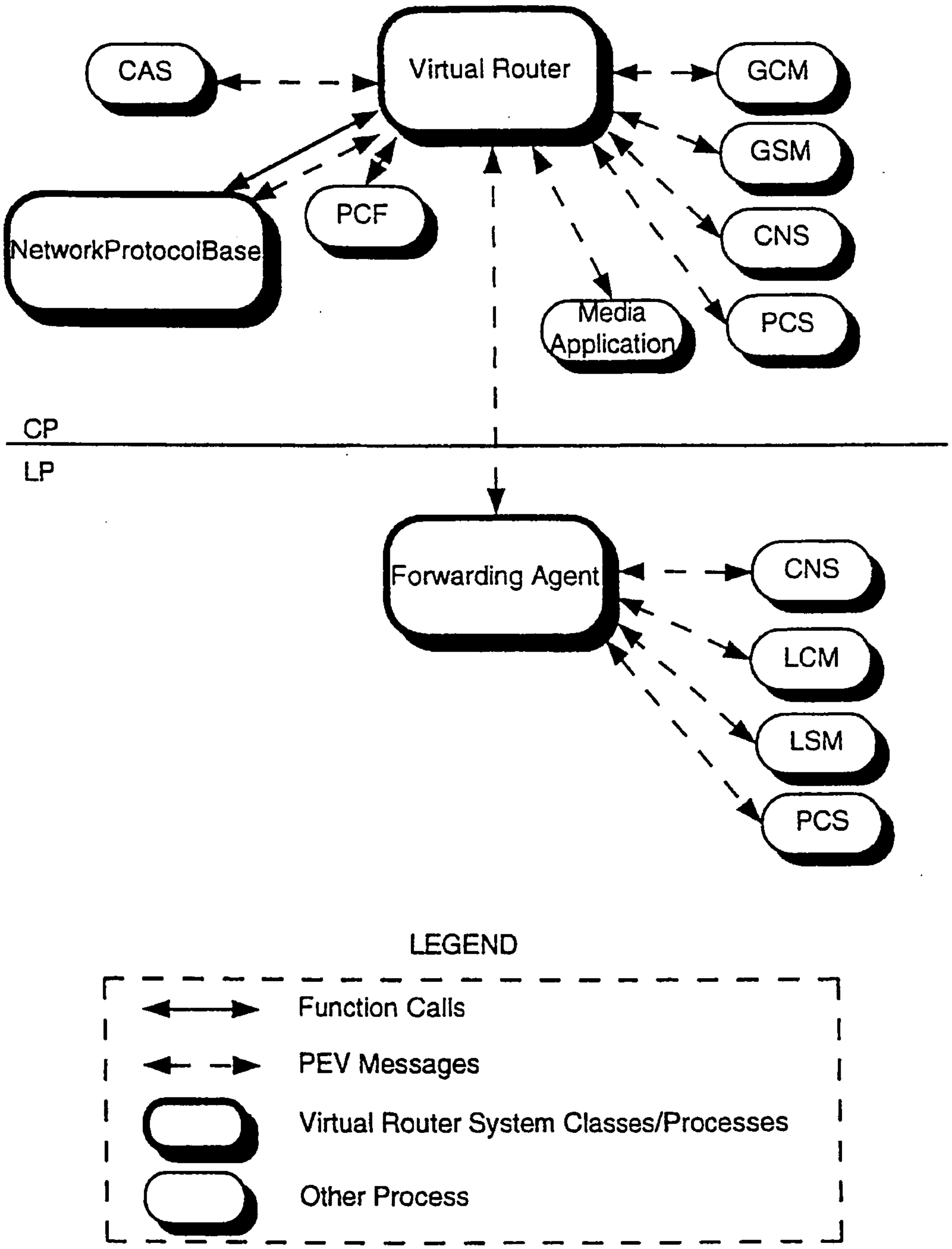

CAS
Virtual Router
GCM
GSM
NetworkProtocolBase
PCF
CNS
Media Application
PCS
CP
LP
Forwarding Agent
CNS
LCM
LSM
PCS
LEGEND
Function Calls
PEV Messages
Virtual Router System Classes/Processes
Other Process

FIG. 22

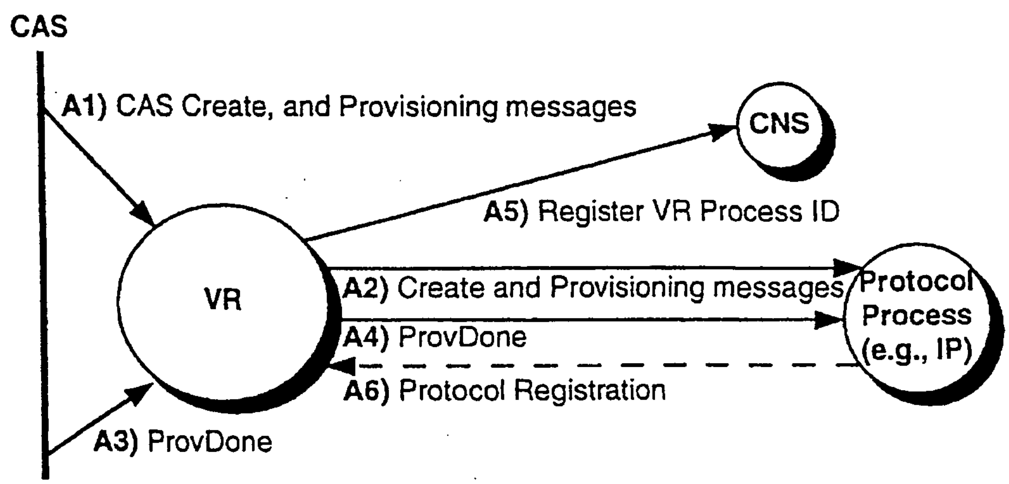

CAS
A1) CAS Create, and Provisioning messages
CNS
A5) Register VR Process ID
VR
A2) Create and Provisioning messages
Protocol Process (e.g., IP)
A4) ProvDone
A6) Protocol Registration
A3) ProvDone

FIG. 23

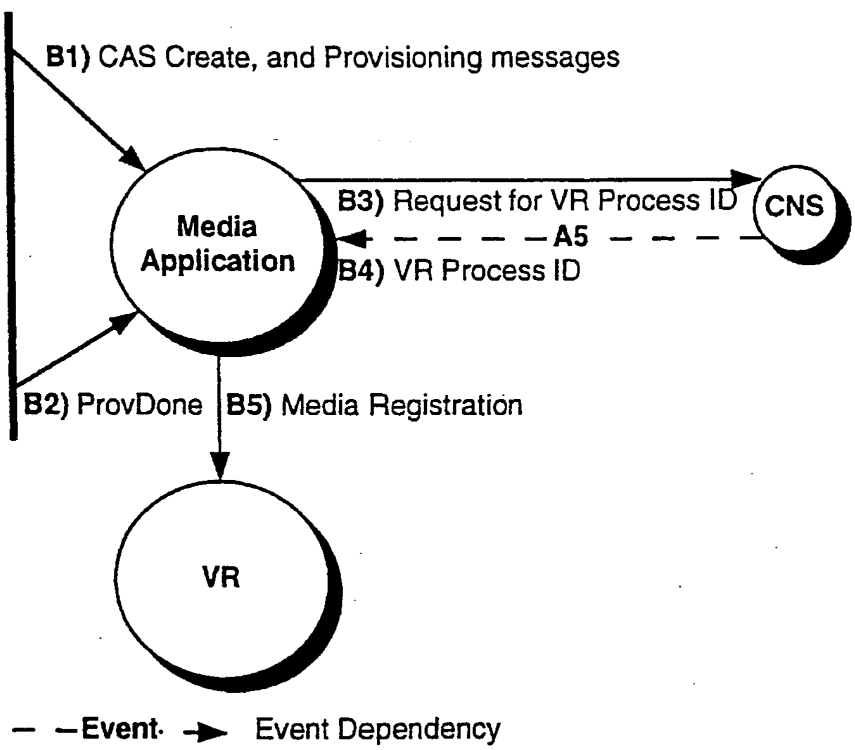

B1) CAS Create, and Provisioning messages
Media Application
B3) Request for VR Process ID
CNS
A5
B4) VR Process ID
B2) ProvDone
B5) Media Registration
VR
Event
Event Dependency

FIG. 24

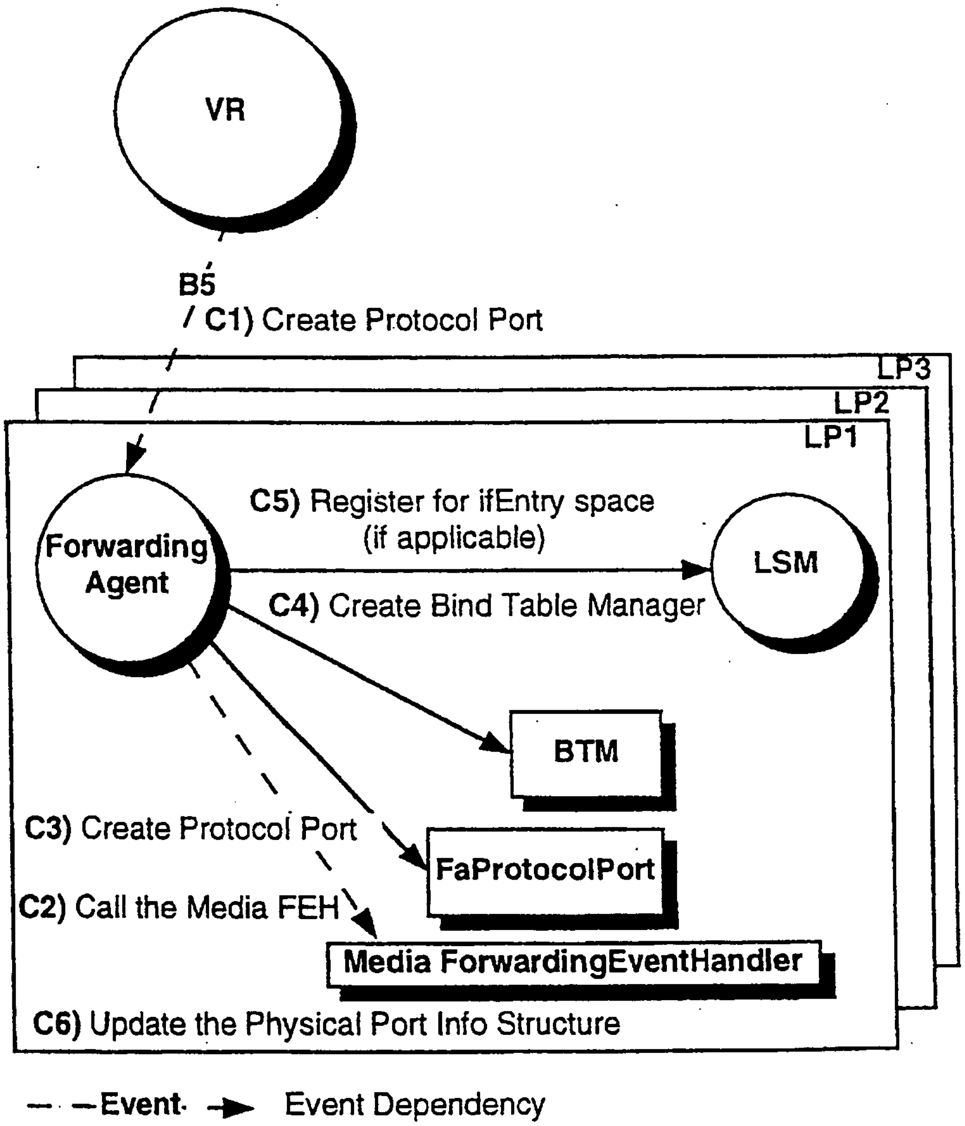

VR
B5
C1) Create Protocol Port
LP3
LP2
LP1
Forwarding Agent
C5) Register for ifEntry space (if applicable)
LSM
C4) Create Bind Table Manager
BTM
C3) Create Protocol Port
FaProtocolPort
C2) Call the Media FEH
Media ForwardingEventHandler
C6) Update the Physical Port Info Structure
Event
Event Dependency

FIG. 25

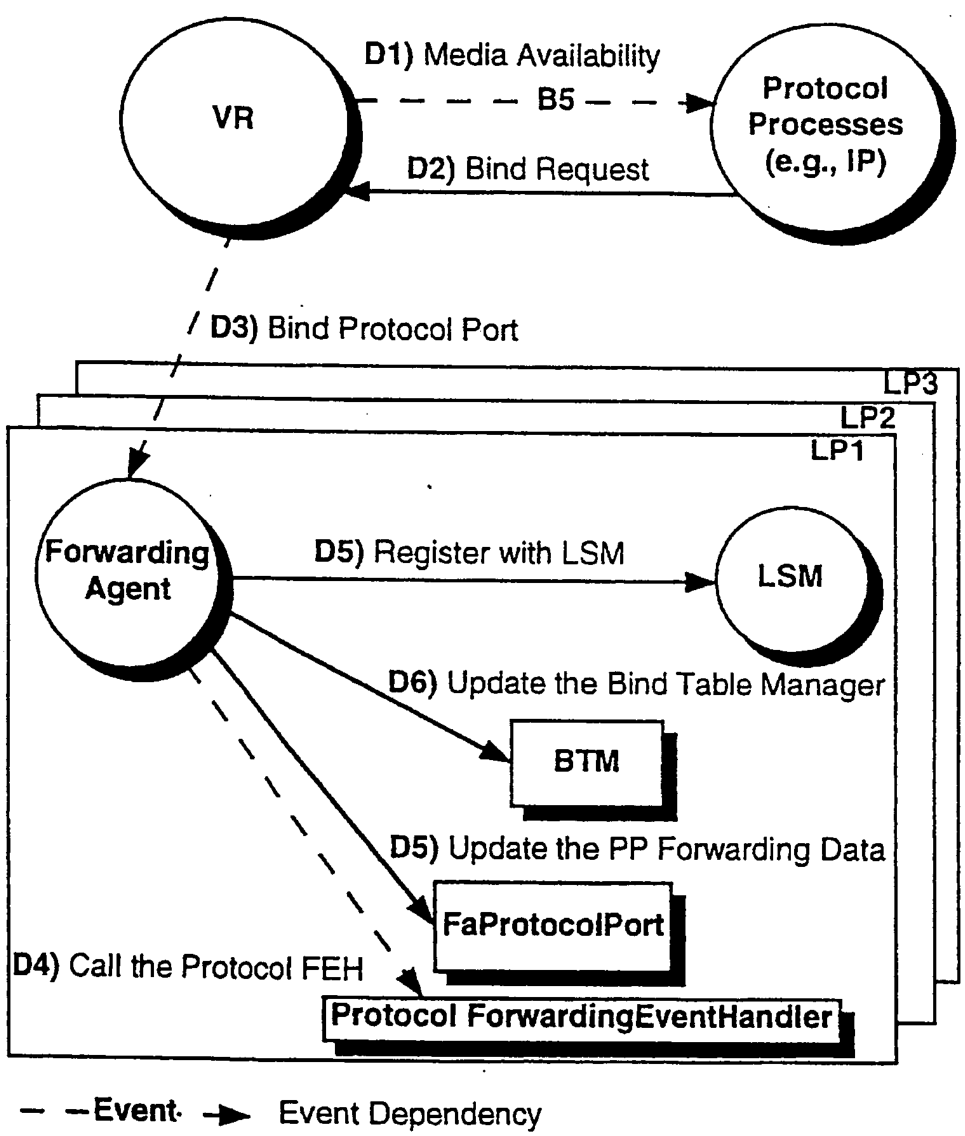

D1) Media Availability
VR
B5
Protocol Processes (e.g., IP)
D2) Bind Request
D3) Bind Protocol Port
LP3
LP2
LP1
Forwarding Agent
D5) Register with LSM
LSM
D6) Update the Bind Table Manager
BTM
D5) Update the PP Forwarding Data
FaProtocolPort
D4) Call the Protocol FEH
Protocol ForwardingEventHandler
Event
Event Dependency

FIG. 26

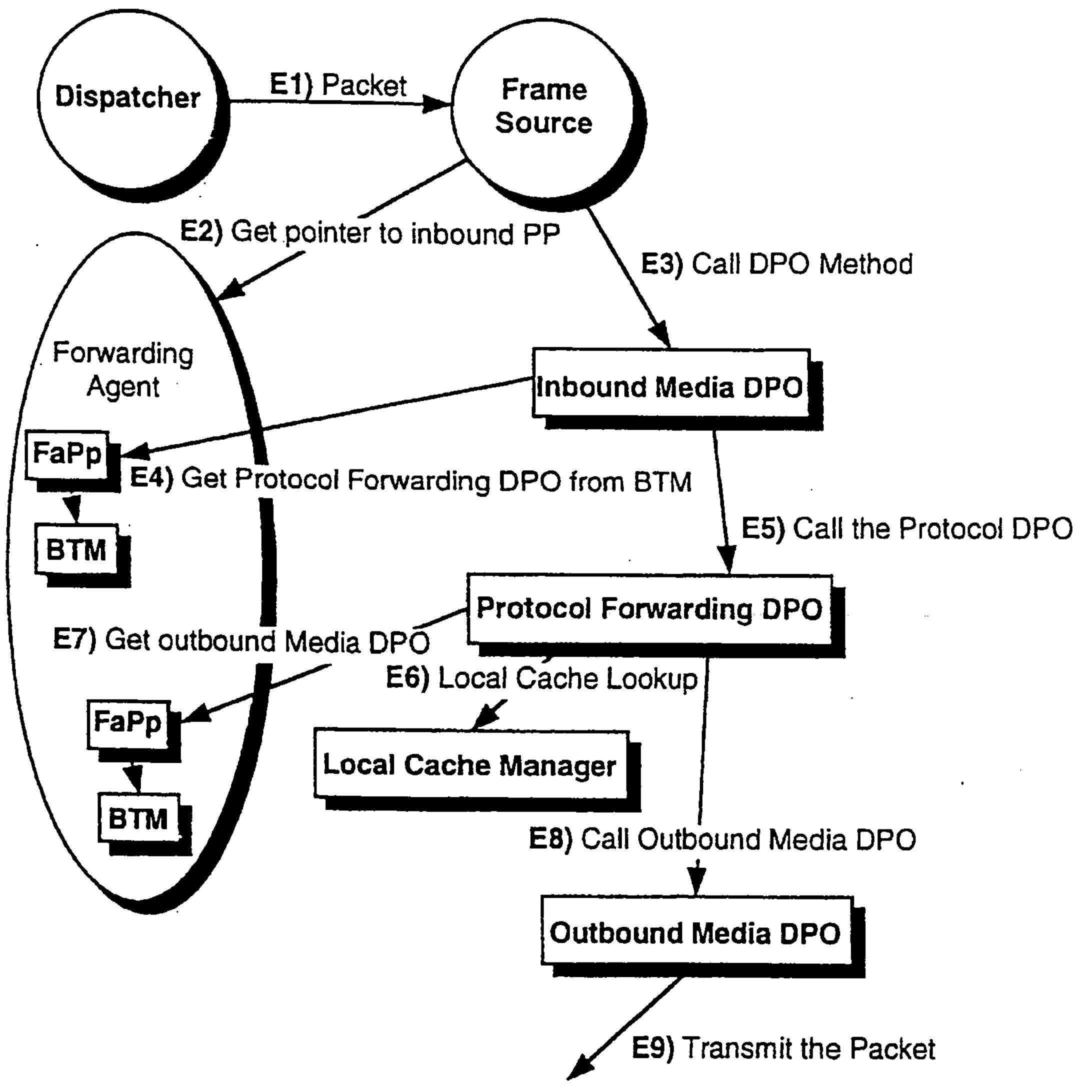

Dispatcher
E1) Packet
Frame Source
E2) Get pointer to inbound PP
E3) Call DPO Method
Forwarding Agent
Inbound Media DPO
FaPp
E4) Get Protocol Forwarding DPO from BTM
BTM
E5) Call the Protocol DPO
Protocol Forwarding DPO
E7) Get outbound Media DPO
E6) Local Cache Lookup
FaPp
Local Cache Manager
BTM
E8) Call Outbound Media DPO
Outbound Media DPO
E9) Transmit the Packet

FIG. 27

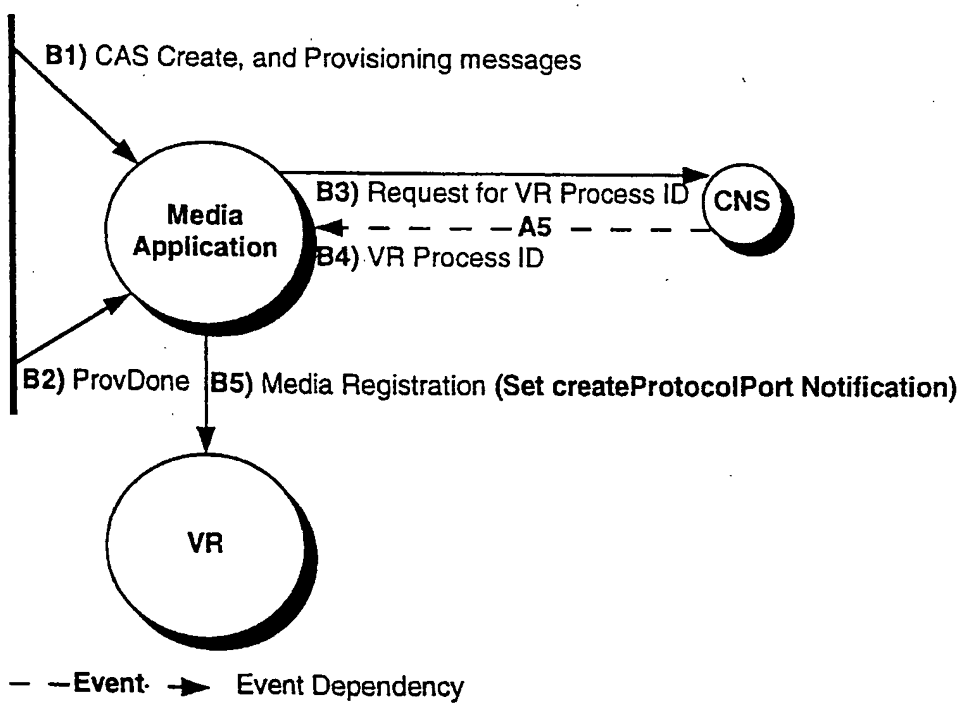

B1) CAS Create, and Provisioning messages
Media Application
B3) Request for VR Process ID
CNS
A5
B4) VR Process ID
B2) ProvDone
B5) Media Registration (Set createProtocolPort Notification)
VR
Event
Event Dependency

FIG. 28

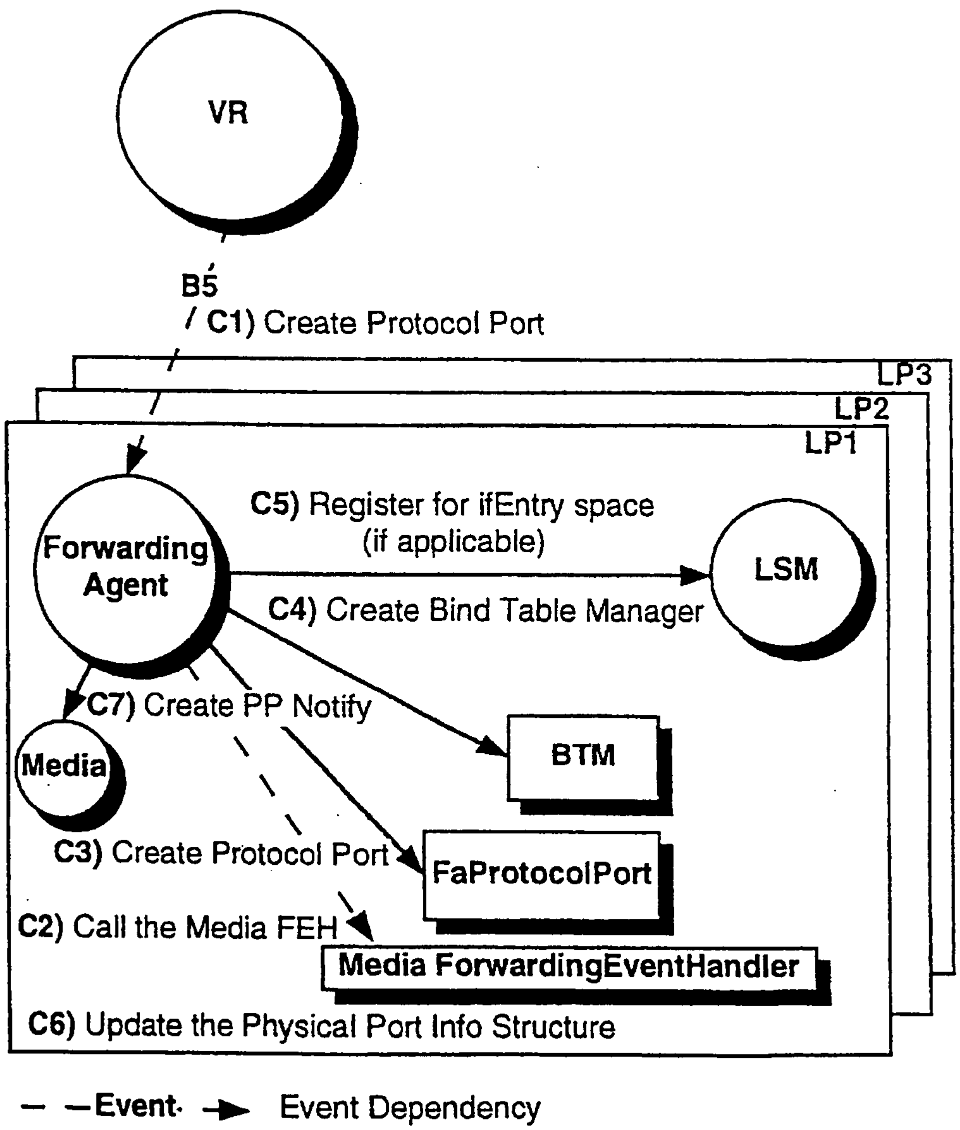

VR
B5
C1) Create Protocol Port
LP3
LP2
LP1
Forwarding Agent
C5) Register for ifEntry space (if applicable)
LSM
C4) Create Bind Table Manager
C7) Create PP Notify
Media
BTM
C3) Create Protocol Port
FaProtocolPort
C2) Call the Media FEH
Media ForwardingEventHandler
C6) Update the Physical Port Info Structure
— —Event
Event Dependency

FIG. 29

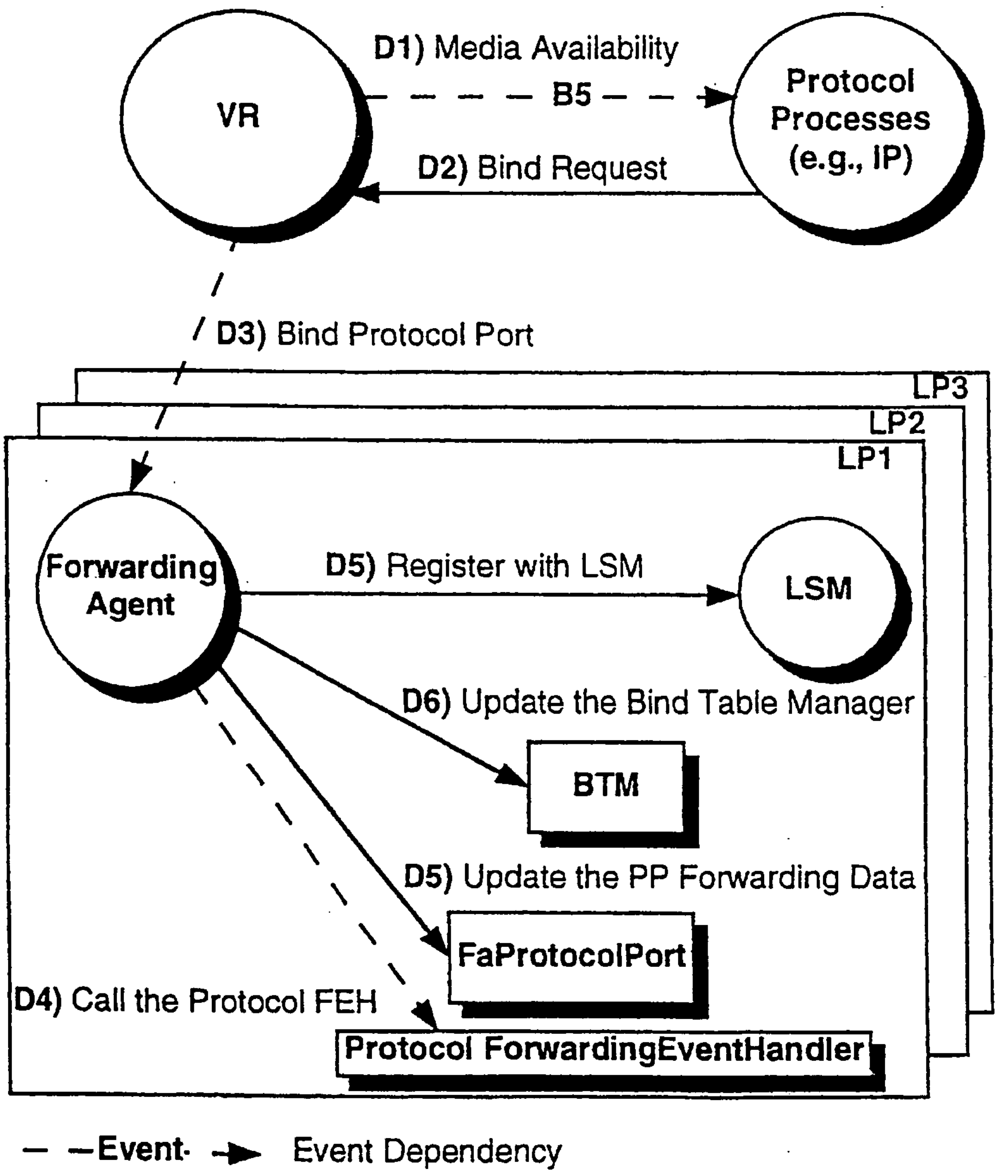

D1) Media Availability
VR
B5
Protocol Processes (e.g., IP)
D2) Bind Request
D3) Bind Protocol Port
LP3
LP2
LP1
Forwarding Agent
D5) Register with LSM
LSM
D6) Update the Bind Table Manager
BTM
D5) Update the PP Forwarding Data
FaProtocolPort
D4) Call the Protocol FEH
Protocol ForwardingEventHandler
Event
Event Dependency

FIG. 30

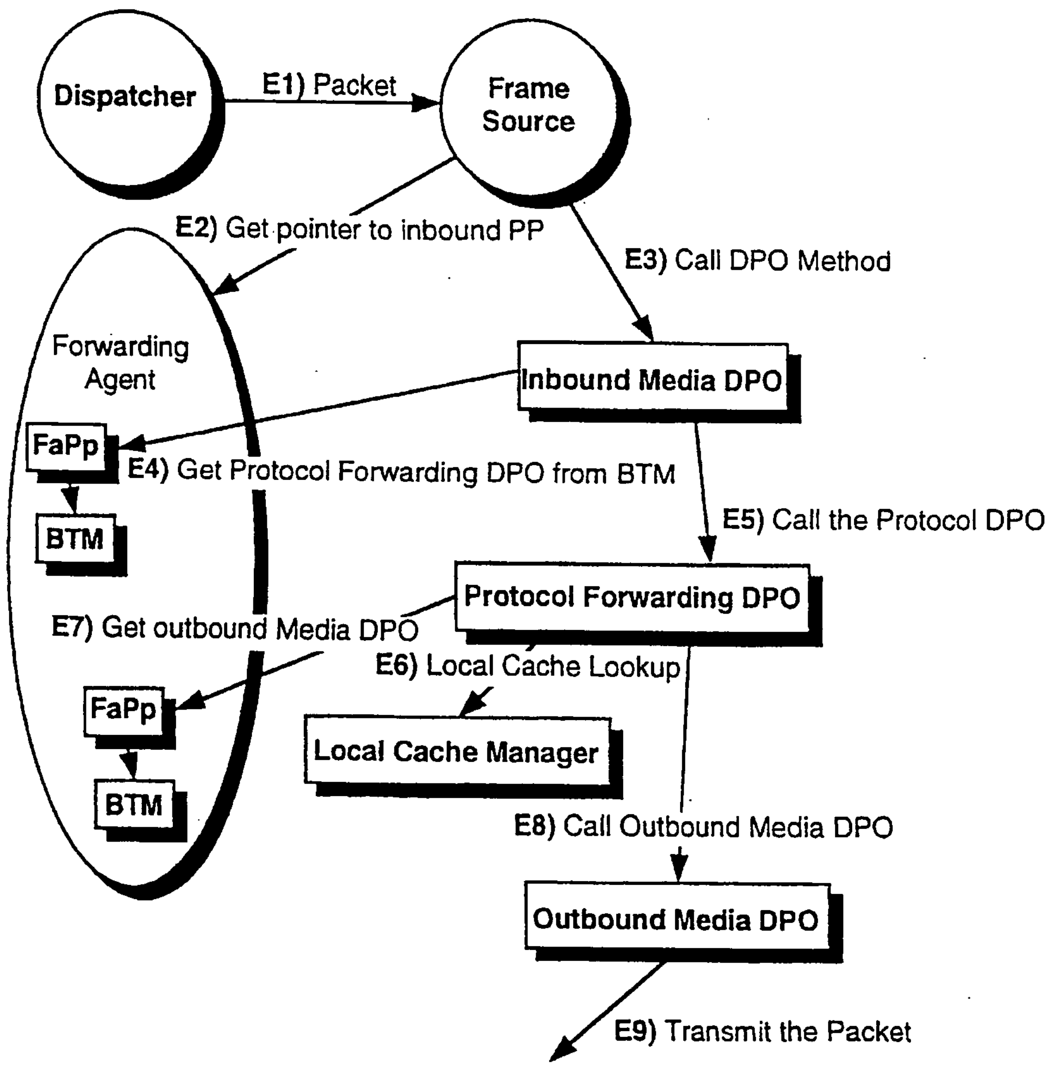

Dispatcher
E1) Packet
Frame Source
E2) Get pointer to inbound PP
E3) Call DPO Method
Forwarding Agent
Inbound Media DPO
FaPp
E4) Get Protocol Forwarding DPO from BTM
BTM
E5) Call the Protocol DPO
Protocol Forwarding DPO
E7) Get outbound Media DPO
E6) Local Cache Lookup
FaPp
Local Cache Manager
BTM
E8) Call Outbound Media DPO
Outbound Media DPO
E9) Transmit the Packet

FIG. 31

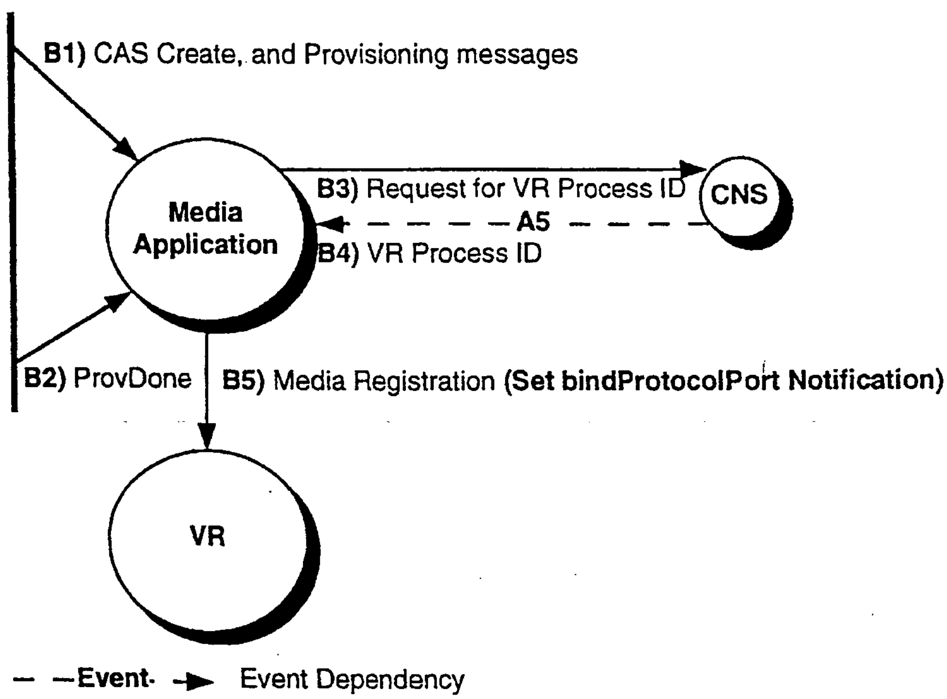

B1) CAS Create, and Provisioning messages
Media Application
B3) Request for VR Process ID
CNS
A5
B4) VR Process ID
B2) ProvDone
B5) Media Registration (Set bindProtocolPort Notification)
VR
Event
Event Dependency

FIG. 32

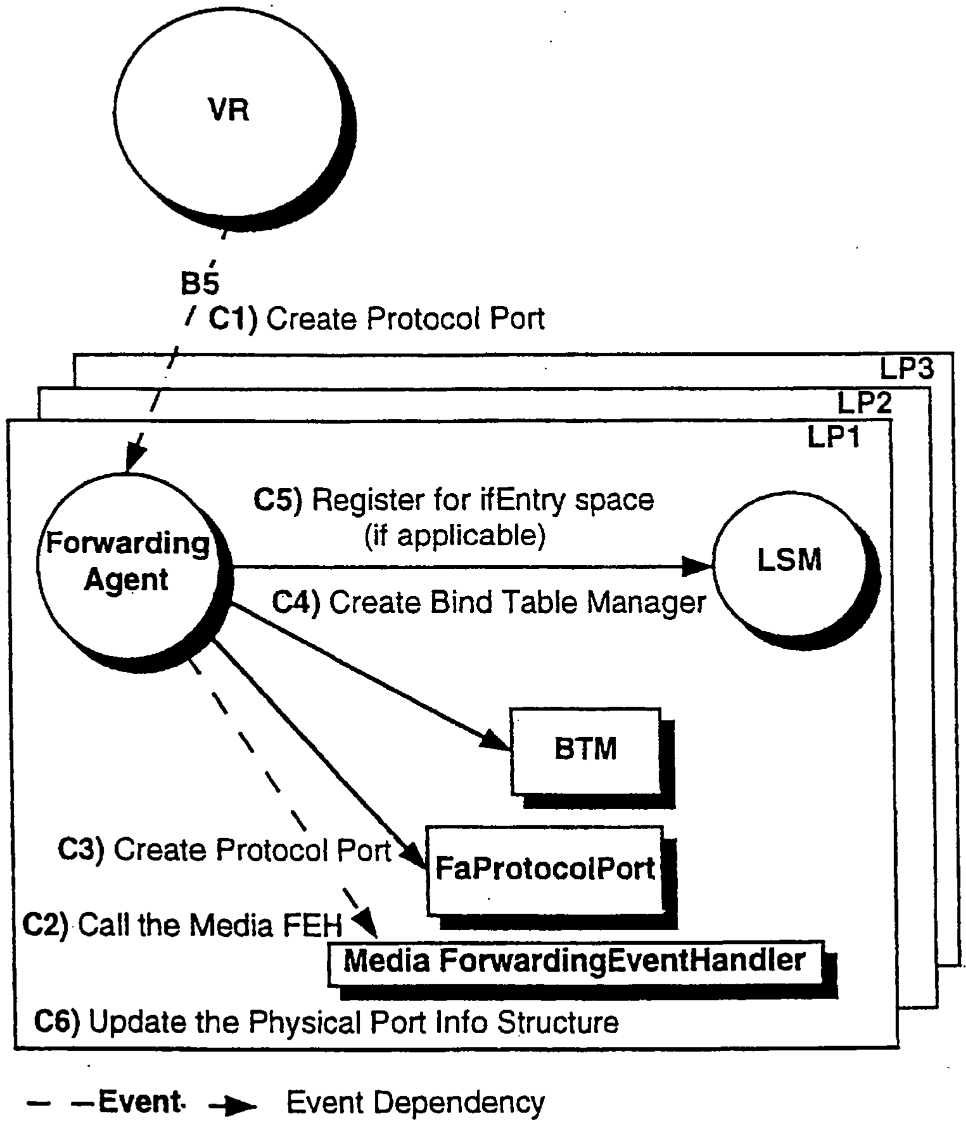

VR
B5
C1) Create Protocol Port
LP3
LP2
LP1
Forwarding Agent
C5) Register for ifEntry space (if applicable)
LSM
C4) Create Bind Table Manager
BTM
C3) Create Protocol Port
FaProtocolPort
C2) Call the Media FEH
Media ForwardingEventHandler
C6) Update the Physical Port Info Structure
Event
Event Dependency

FIG. 33

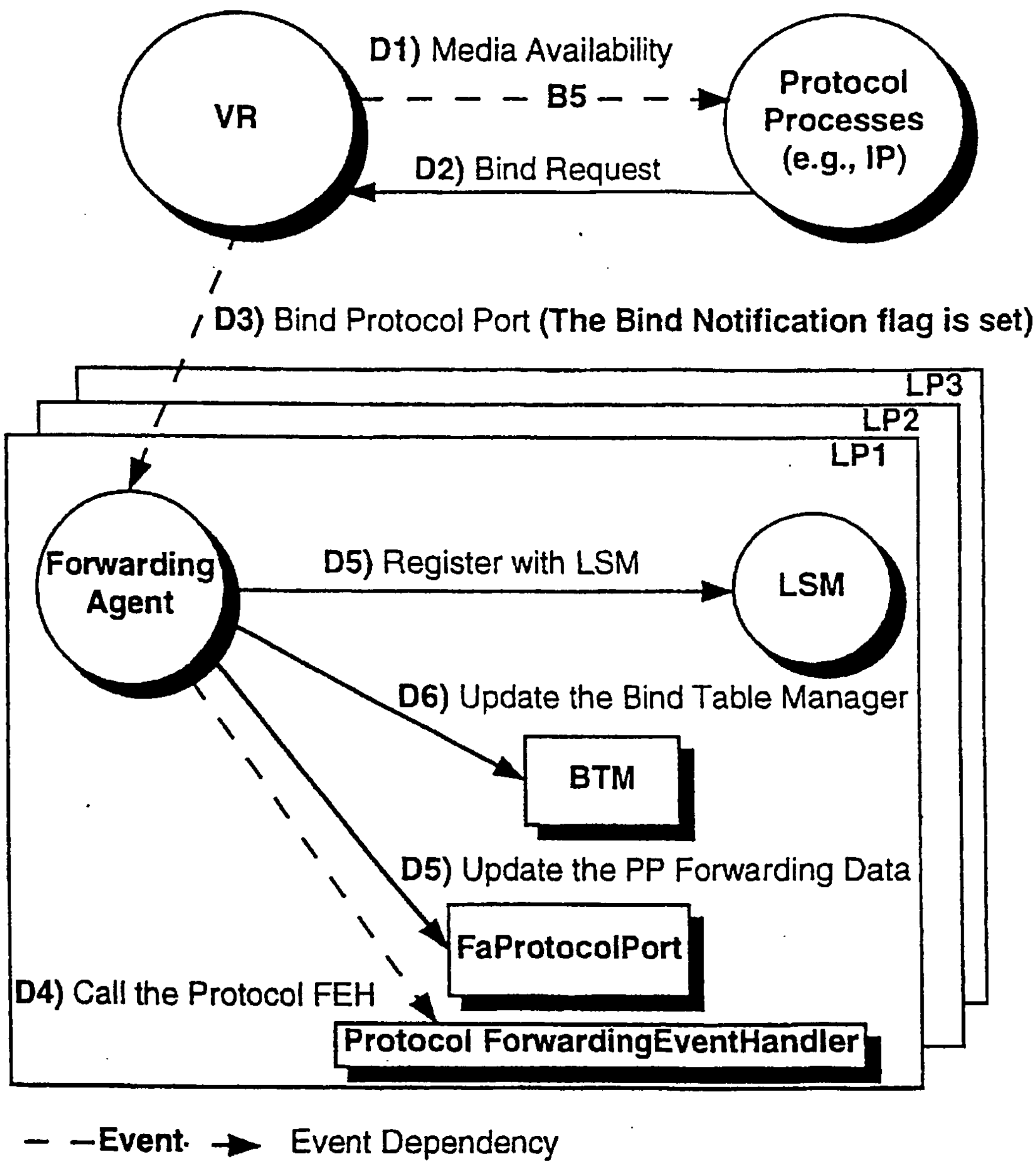

D1) Media Availability
B5
VR
Protocol Processes (e.g., IP)
D2) Bind Request
D3) Bind Protocol Port (The Bind Notification flag is set)
LP3
LP2
LP1
Forwarding Agent
D5) Register with LSM
LSM
D6) Update the Bind Table Manager
BTM
D5) Update the PP Forwarding Data
FaProtocolPort
D4) Call the Protocol FEH
Protocol ForwardingEventHandler
— —Event
Event Dependency

FIG. 34

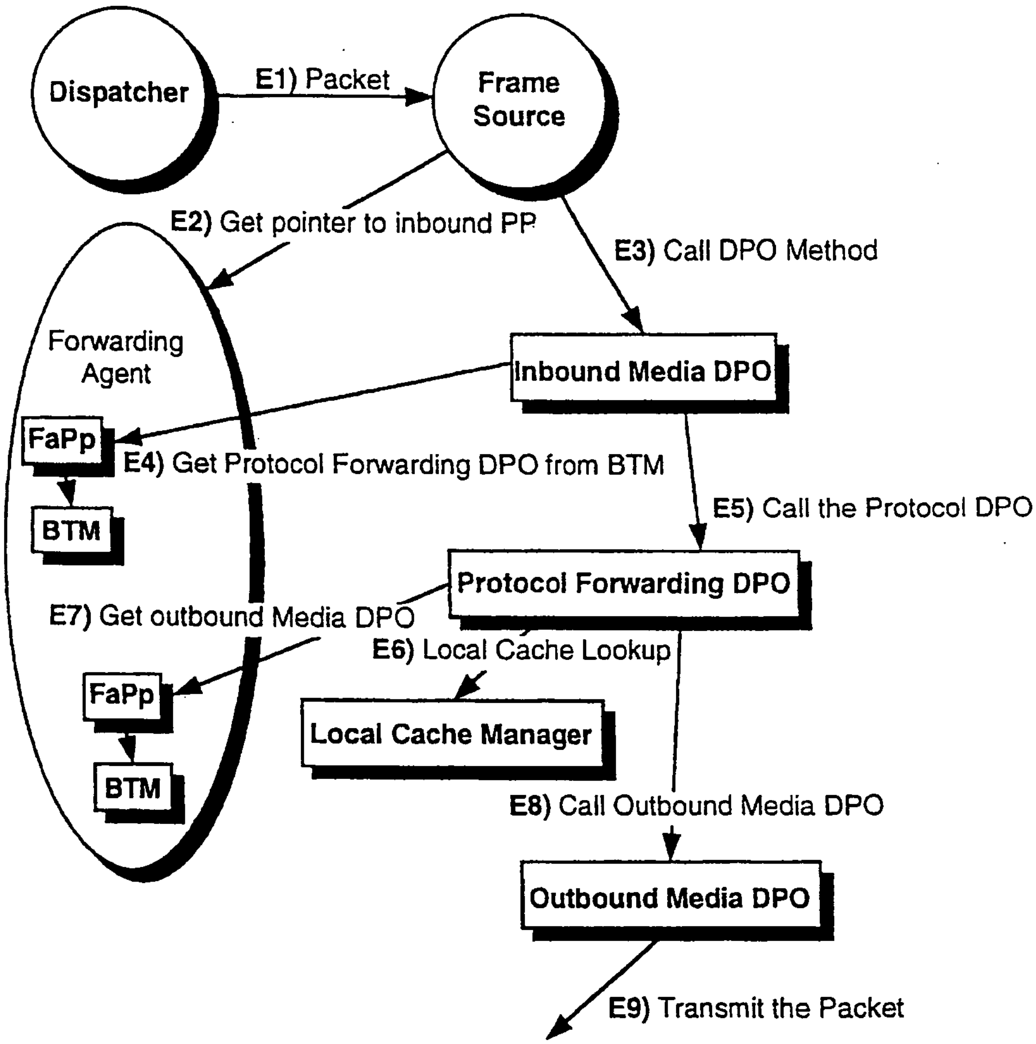
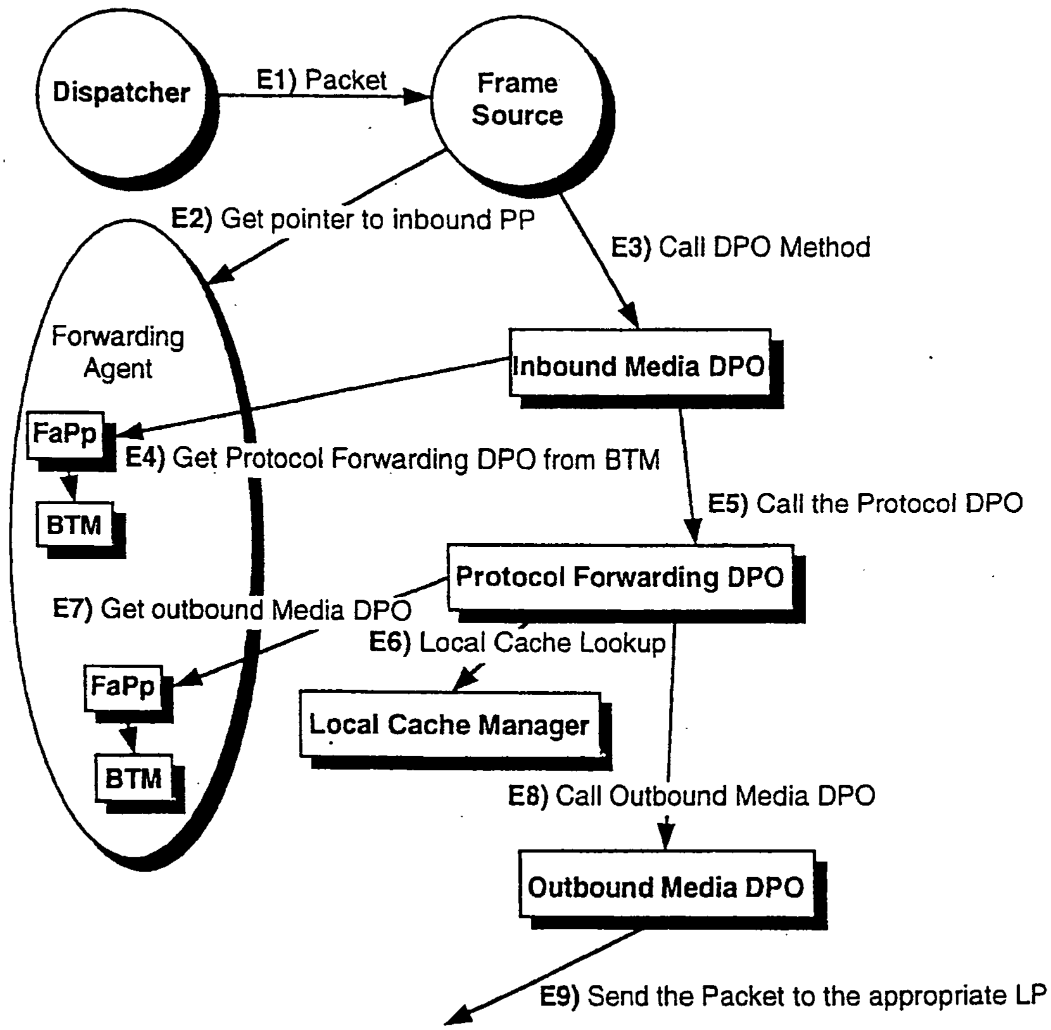

Dispatcher
E1) Packet
Frame Source
E2) Get pointer to inbound PP
E3) Call DPO Method
Forwarding Agent
Inbound Media DPO
FaPp
E4) Get Protocol Forwarding DPO from BTM
BTM
E5) Call the Protocol DPO
Protocol Forwarding DPO
E7) Get outbound Media DPO
E6) Local Cache Lookup
FaPp
Local Cache Manager
BTM
E8) Call Outbound Media DPO
Outbound Media DPO
E9) Transmit the Packet

FIG. 35



Dispatcher
E1) Packet
Frame Source
E2) Get pointer to inbound PP
E3) Call DPO Method
Forwarding Agent
Inbound Media DPO
FaPp
E4) Get Protocol Forwarding DPO from BTM
BTM
E5) Call the Protocol DPO
Protocol Forwarding DPO
E7) Get outbound Media DPO
E6) Local Cache Lookup
FaPp
Local Cache Manager
BTM
E8) Call Outbound Media DPO
Outbound Media DPO
E9) Send the Packet to the appropriate LP

FIG. 36

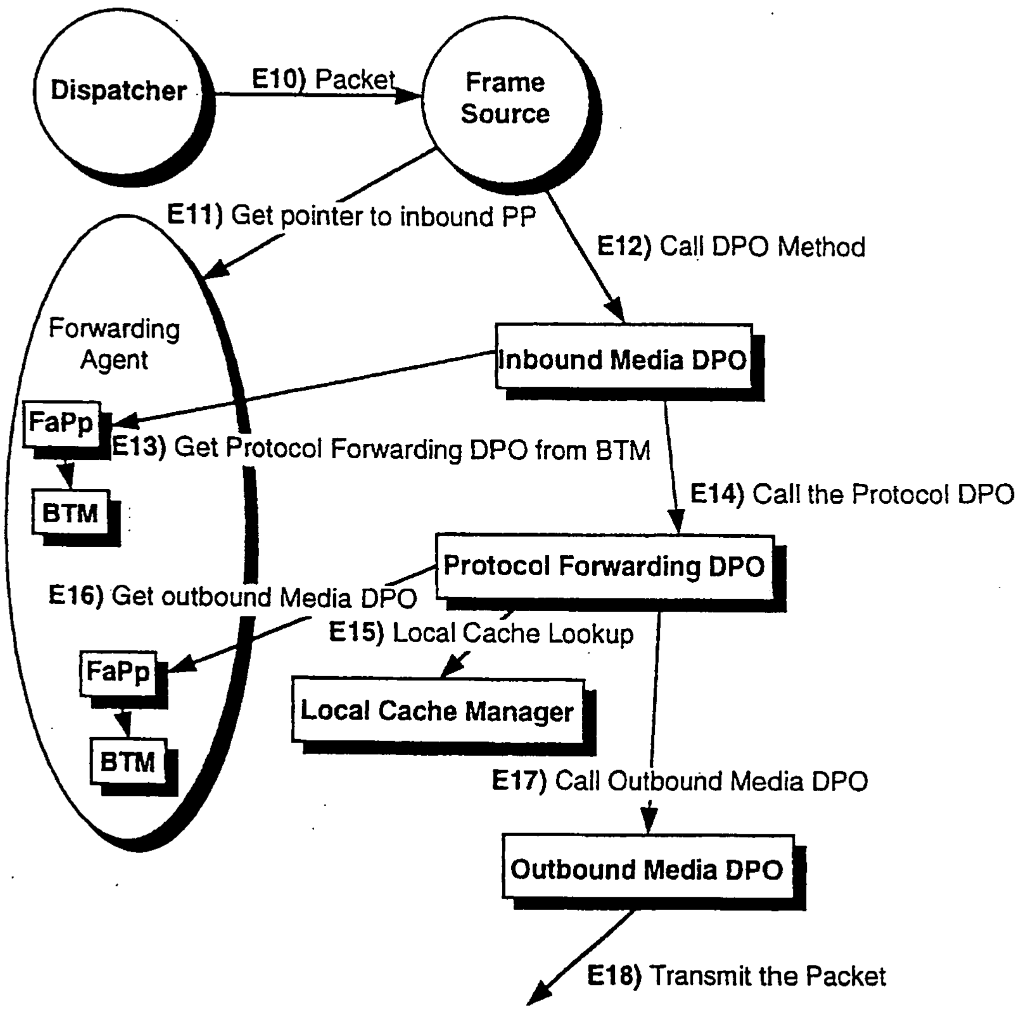

Dispatcher
E10) Packet
Frame Source
E11) Get pointer to inbound PP
E12) Call DPO Method
Forwarding Agent
Inbound Media DPO
FaPp
E13) Get Protocol Forwarding DPO from BTM
BTM
E14) Call the Protocol DPO
Protocol Forwarding DPO
E16) Get outbound Media DPO
E15) Local Cache Lookup
FaPp
Local Cache Manager
BTM
E17) Call Outbound Media DPO
Outbound Media DPO
E18) Transmit the Packet

FIG. 37

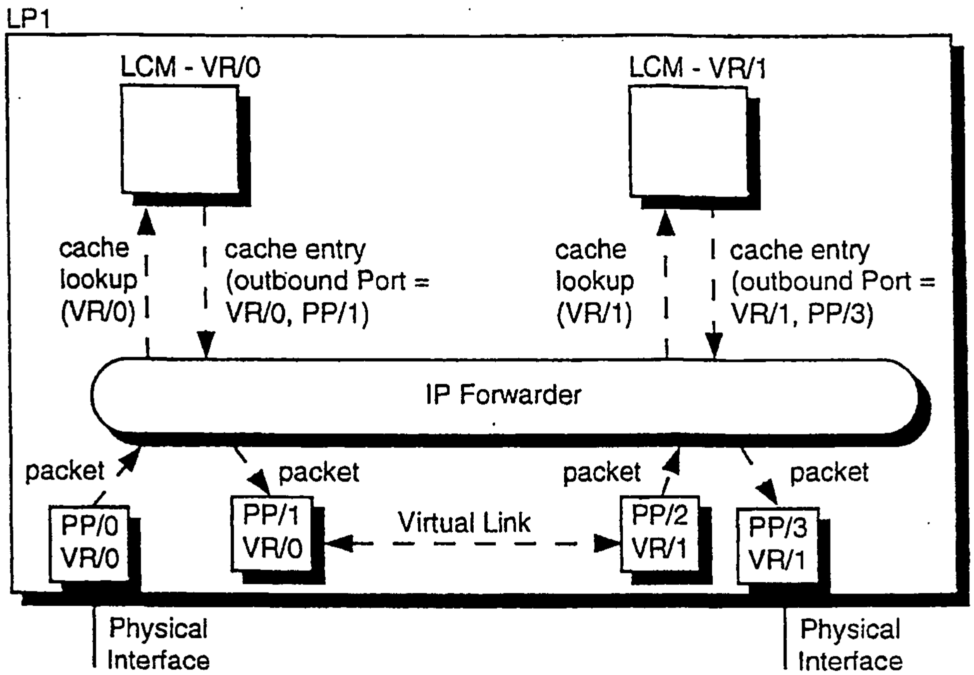

LP1
LCM - VR/0
LCM - VR/1
cache lookup (VR/0)
cache entry (outbound Port = VR/0, PP/1)
cache lookup (VR/1)
cache entry (outbound Port = VR/1, PP/3)
IP Forwarder
packet
packet
packet
packet
PP/0 VR/0
PP/1 VR/0
Virtual Link
PP/2 VR/1
PP/3 VR/1
Physical Interface
Physical Interface

FIG. 38

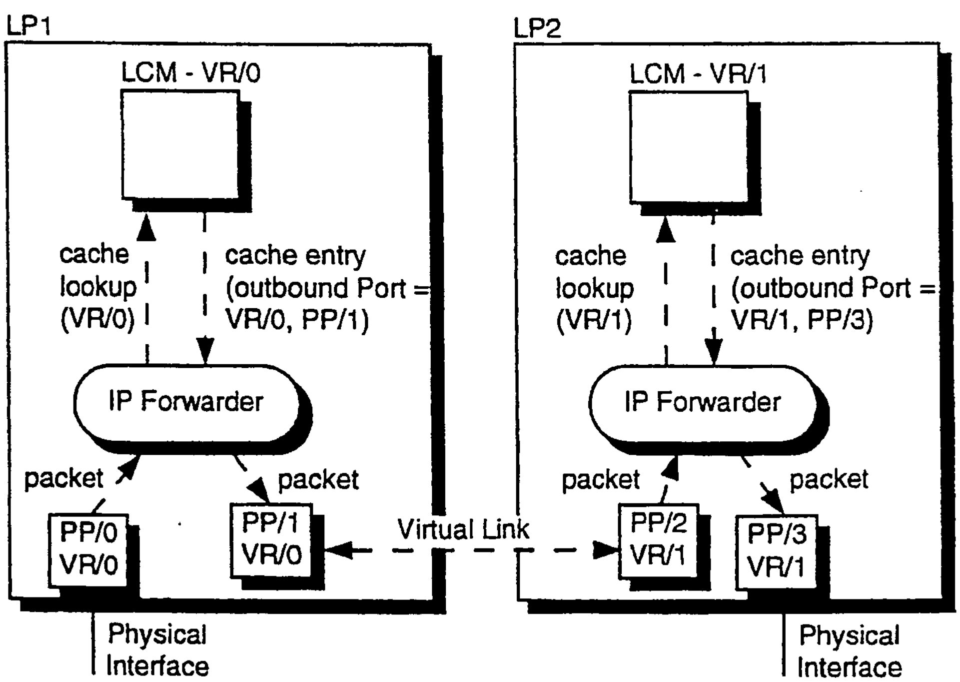

LP1
LP2
LCM - VR/0
LCM - VR/1
cache lookup (VR/0)
cache entry (outbound Port = VR/0, PP/1)
cache lookup (VR/1)
cache entry (outbound Port = VR/1, PP/3)
IP Forwarder
IP Forwarder
packet
packet
packet
packet
PP/0 VR/0
PP/1 VR/0
Virtual Link
PP/2 VR/1
PP/3 VR/1
Physical Interface
Physical Interface

FIG. 39

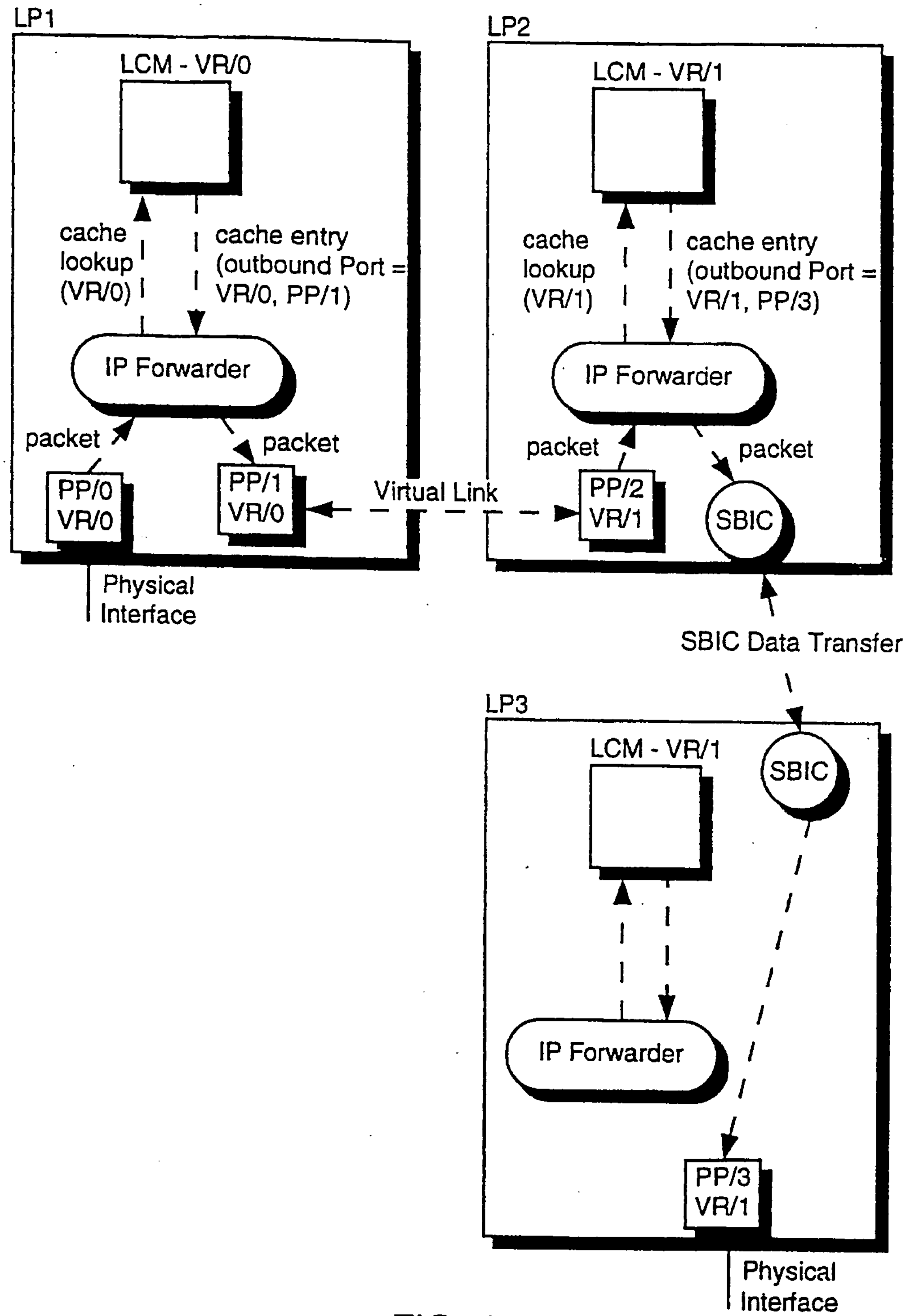

LP1
LCM - VR/0
cache lookup (VR/0)
cache entry (outbound Port = VR/0, PP/1)
IP Forwarder
packet
packet
PP/0 VR/0
PP/1 VR/0
Physical Interface
Virtual Link
LP2
LCM - VR/1
cache lookup (VR/1)
cache entry (outbound Port = VR/1, PP/3)
IP Forwarder
packet
packet
PP/2 VR/1
SBIC
SBIC Data Transfer
LP3
LCM - VR/1
SBIC
IP Forwarder
PP/3 VR/1
Physical Interface

FIG. 40

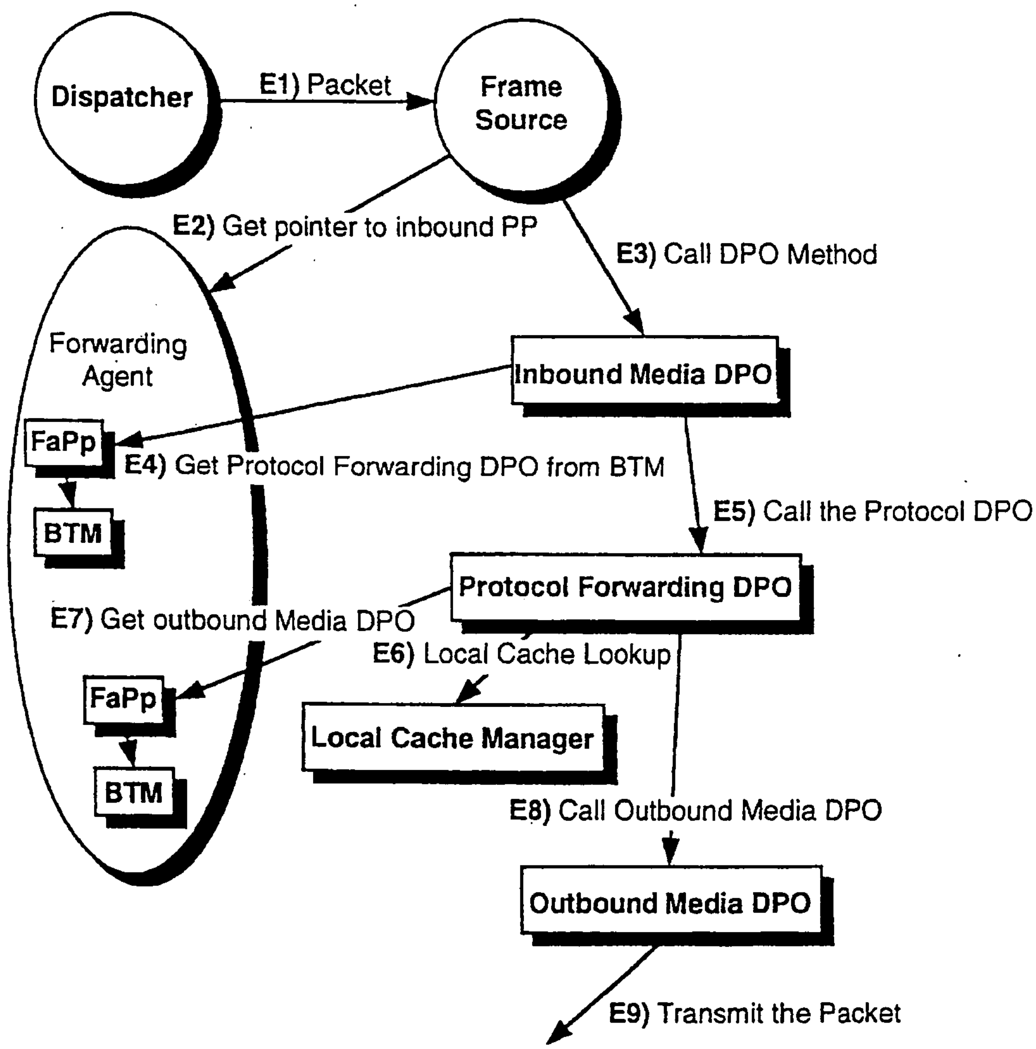

Dispatcher
E1) Packet
Frame Source
E2) Get pointer to inbound PP
E3) Call DPO Method
Forwarding Agent
Inbound Media DPO
FaPp
E4) Get Protocol Forwarding DPO from BTM
BTM
E5) Call the Protocol DPO
Protocol Forwarding DPO
E7) Get outbound Media DPO
E6) Local Cache Lookup
FaPp
Local Cache Manager
BTM
E8) Call Outbound Media DPO
Outbound Media DPO
E9) Transmit the Packet

FIG. 41

METHOD AND APPARATUS FOR VIRTUAL SWITCHING

RELATED INVENTIONS

The present invention is related to:

Co-pending U.S. patent application Ser. No. 08/366,221, filed on Dec. 29, 1994, which is entitled "Method And Apparatus For Accelerated Packet Forwarding" by Mark Bakke et al.,

Co-pending U.S. patent application Ser. No. 08/366,225, filed on Dec. 29, 1994, which is entitled "Method And Apparatus For Accelerated Packet Processing" by Geof Stone,

Co-pending U.S. patent application Ser. No. 08/366,222, filed on Dec. 29, 1994, which is entitled "Method And Apparatus For Radix Decision Packet Processing" by Geof Stone,

and which were all filed concurrently herewith and assigned to the assignee of the present invention.

FIELD OF THE INVENTION

The present invention relates generally to data communication networks. More particularly, the present invention relates to the operation of virtual switches within physical switching systems that direct the flow of protocol data units in the data communication networks.

BACKGROUND OF THE INVENTION

In a data communication network, a forwarding device (e.g., a data packet switch) directs protocol data units (e.g., data packets) from one network node to another. These data packets may include voice, video, or data information as well as any combination thereof.

To better understand how forwarding devices work within a data communication network, an analogy may be helpful. In many respects, data communication networks are similar to postal delivery systems, with pieces of mail, such as letters or packages, being comparable to the data packets which are transferred within a data communication network. In a postal delivery system, the pieces of mail may be input into the postal delivery system in a variety of ways. Once within the postal delivery system, all of the pieces of mail are collected and transported to nearby processing facilities where the pieces of mail are sorted for further processing. Although each piece of mail will have a unique delivery address, most of the pieces of mail are automatically sorted by a shorter zip code or some other type of routing code. Letters without zip codes must be sorted and processed by hand. Some postal delivery systems also have special forms of encoded delivery addresses, such as Post Office box numbers at a Post Office, which are not recognizable by other postal delivery systems such as Federal Express or United Parcel Service. Regardless of which particular postal delivery system the piece of mail is deposited into, once the mail has been sorted by destination it is routed through additional intermediary processing facilities until it arrives at the local indicated by the destination on the piece of mail. At this point, the zip code or routing code is no longer sufficient to deliver the piece of mail to the intended destination and the local delivery office must further decode the destination address in order to deliver the piece of mail to the intended recipient. In addition to processing pieces of mail for routing the mail to the correct destination, the pieces of mail may go on through several other processing steps. For example, if the piece of mail is going out of the country, it must go through a customs operation in each country. If the national postal delivery system is being used to deliver the piece of mail then it must also be transferred from one national postal delivery system to another. In a private postal delivery system however, this transfer step would not be necessary. The pieces of mail may also be monitored or filtered for such things as mail fraud violation or shipment of hazardous materials.

Data packets are manipulated in a data communication network in a manner similar to that by which pieces of mail are delivered in a postal delivery system. Data packets, for example, are generated by many different types of means and are placed onto a communication network. Typically, the data packets are concentrated into a forwarding device, such as a local bridge or router, and are then directed by size and destination over one or more media types (e.g., fiber optic) which are connected to further forwarding devices that could be other larger or smaller bridges or routers. These destination devices then deliver the data packet to its terminal end point (i.e., the end user). Along the way the data communication network may perform filtering and monitoring functions with respect to the data packets.

Just like postal delivery systems have experienced ever increasing volumes of mail which must be delivered, the volume of protocol data units being transferred across computer networks continues to increase as experience is being gained with this new form of communication delivery system and as more and more applications, with more and more expansive communications requirements are being developed. In addition, quickly changing technology has made the underlying data transmission resources for computer communication networks relatively inexpensive. Fiber optics, for example, offer data transfer rates in the gigabyte per second range.

One of the existing types of forwarding devices which offer the greatest potential to meet the increasing demand on throughput rates are packet switches. Several classes of packet switches exist. Each class differs substantially from the other class of devices, but all may be commonly referred to as packet switches or forwarding devices.

A first class of packet switches is that commonly used in digital telephone exchanges. By analogy, these switches can perform the functions only of a dedicated mail truck which relays mail between post offices and drops mail pouches on a post office loading dock. These switches are intended only to transfer packets among the devices in a single station, such as a telephone exchange, and are not capable of performing any sorting operations. The format of the packet in these systems is chosen to make the hardware in the switch as simple as possible; and this usually means that the packets include fields designed for direct use by the hardware. The capabilities of this class of switches (for example, in such areas as congestion control) are very limited in order to keep the hardware simple.

A second class of packet switches is used in smaller or restricted computer networks, such as X.25 networks. By analogy, these switches are equivalent to a group of #10 envelope sorters in the Post Office. These sorters handle and process this size envelope efficiently within the post office by performing limited sorting and routing functions, but can not by themselves deliver mail to its destination. In some sense, these switches are very different from the first class of packet switches described above, because several of this second class of packet switches can work together like several #10 envelope sorters can work at one time in the Post

Office. However, there is one substantial similarity in that this second class of switches can only handle one format of packets (i.e., the protocols). The formats handled by the second class of packet switches is much more complex than those in the first class. This greater complexity is necessary because the protocols are designed to work in less restricted environments, and because the packet switches must provide a greater range of services. While the formats interpreted by the first class of switches are chosen for easy implementation in hardware, the data packets handled by this second class of switches are generally intended to be interpreted by software (which can easily and economically handle the greater complexity) and provides the inherit benefit of incremental flexibility in the design of the packet switch.

In a third class of packet switches, the packet protocols are intended to be used in very large data networks having many very dissimilar links (such as a mix of very high speed local area networks (LANs) and low speed long distance point to point lines). Examples of such protocols are the United States designed Transmission Control Protocol/Internet Protocol (TCP/IP), and the International Standards Organization's Connectionless Network Protocol (CLNP) protocols.

In addition, this third class of switches (commonly referred to as bridge/routers) often must handle multiple protocols simultaneously. This third class of switches is very similar to the mail processing devices used in the modern postal system. Just as there are many countries, there are many data packet protocols used in computer networks. While a single postal system was once thought to be sufficient to handle mail going anywhere in the world, today several competing systems like United Parcel Service, Federal Express, and the U.S. Postal Service exist to handle the special needs of mail going to every country, state, city, town, and street in the world. Similarly, in computer communication systems, the packet switches are more involved in the carrying of data, and must understand some of the details of each protocol to be able to correctly handle data packets which are being conveyed in that protocol. The routers in this third class of packet switches often have to make fairly complex changes to the data packets as they pass through the packet switch.

It is this latter class of packet switches to which the following detailed description primarily relates. It will be appreciated however, that the detailed description of this invention can readily be applied to the first and second class of switches as well.

In current conventional packet switch design, a programmed general purpose processor examines each data packet as it arrives over the network interface and then processes that packet. Packet processing requires assignment of the data packet to an outbound network interface for transmission over the next communications link in the data path.

Currently, most bridge/router implementations rely heavily on off-the-shelf microprocessors to perform the packet forwarding functions. The best implementations are able to sustain processing rates approaching 100,000 packets per second (PPS). When dealing with media such as Ethernet or current telecommunications lines, this processing rate is more than adequate. When faster media such as the Fiber Distributed Data Interface (FDDI) are used, existing processing rates may still be sufficient as long as there is only one such high packet rate interface present. When multiple high packet rate interfaces are used, 100,000 PPS become inadequate. Current software-based implementations for bridges/routers are simply not capable of media-rate packet forwarding on emerging media such as asynchronous transfer mode (ATM) or Optical Connection-12 Synchronous Optical Network (OC-12 SONET) which can accommodate communication rates up to 6 times the current 100 megabits per second limits to rates of 600 megabits per second. It should be noted that the ever increasing power of off-the-shelf microprocessors might solve the throughput problem, but this is probably a vain hope. For example, a single OC-24 ATM interface can sustain nearly 3 million internetworking protocol (IP) packets per second. This is over 30 times the rates achieved by the current best software techniques. If processing power doubles every year, the wait for sufficient processing power to make a software approach viable would be at least 4–5 years. In addition, the media capabilities will likely continue to increase over such a span of years. Additionally, any such processor will likely require large amounts of the fastest (most expensive) memory available to operate at full speed, resulting in an unacceptably high system cost.

Fortunately most individual packet switch customers will never require sustained packet transfer rates at these levels. However, the traditional approach of individual customers purchasing routers, bridges, modems, and leased phone lines is changing. A trend towards developing Metropolitan Area Networks (MANs) is beginning in the networking industry as an alternative to the traditional approach of individual customer local area networks (LANs) connected through customer owned leased telecommunication lines.

The more successful entrants in this area are capitalizing on three trends:

Fiber optic cable can be laid to most business and industrial premises by organizations possessing rights of way; this cable can be used to carry 100 Megabits/second or more of customer traffic, a bandwidth that appears almost limitless to customers.

The "demarkation point" is changing from a pair of copper wires to an Ethernet socket; the MAN vendor takes responsibility for the delivery of Ethernet packets between sites specified by the customer. The customer does not have to be concerned with the intricacies of bridges, routers, and modems, which permits market penetration into a far less sophisticated customer base.

Most potential customers are not interested in a public network connection. They simply want to interconnect a number of buildings or divisions which constitute the customer's enterprise in a metropolitan area.

These MAN vendors are dealing with "customers" in the truest sense of the word, where customer and MAN vendor are independent enterprises. The trends towards corporate decentralization are even producing analogous situations within large enterprises.

Second, enterprises are becoming far more distributed than before, and the very definition of an "enterprise" is changing. Where in the 1980's all individuals involved in a program could be expected to reside in one or two well defined locations, a more modern "enterprise" may consist of individuals from several divisions, several corporations, consultants, roving sales and marketing people, and workers who want to telecommute at their convenience. At the same time, this modern enterprise needs to protect their information from disclosure or sabotage from without the group while preserving a liberal access policy from within.

A wide area "backbone" is a tremendous investment on the part of any large enterprise. Yet at the same time, host computers and small scale networks are becoming easier to

administer while the expertise to administer them becomes more widespread. At the same time, organizations with a bias towards decentralization are seeing departments and divisions owning "their" hosts and "their" networks that they want to plug into a wide area backbone in order to carry their traffic. This traffic typically consists of communications to other divisions; however, increasingly it will also consist of traffic within a division with widely scattered sites.

All of this follows a known trend of increasing decentralization in the workplace. Many years ago, Management Information System (MIS) computers and all the networks in the enterprise. Access policy (such as was needed then) could largely be done through system administration of the host computers.

The advent of personal computers and affordable workstations meant that the networking administrators no longer owned all of the host computers anymore, yet these same MIS organizations are still charged with their traditional role of ensuring the integrity of the enterprise's data. This has led to the rise of routing and filtering functions within routers, making access control, a network, rather than a host problem.

Now the networking industry is moving up one more level. Today, clients not only own their own hosts, they own their own networks and want to connect these networks on a network to network basis. Yet at the same time, the need to preserve the integrity of data moving among client networks still exists. This trend is producing not just a "network", but a "network of networks", where the purpose of a backbone is to serve the needs and foibles of its constituent networks, not all of which may belong to the same enterprise.

The concept of a "network of networks" is not new. In fact, this was one of the guiding philosophies which led to the original creation of the Internet. Unfortunately, the logic to support this has only been applied to Internet Protocol and more recently to the Open Systems Interconnection (OSI) model. IP has been designed to perform this trick once (at the Internet level) and is little help in organizing traffic within a single IP network. Furthermore, IP cannot cope with the notion that a single network may be scattered at different points throughout the Internet.

Thus, a need exists for a way to provide equivalent protocols and management tools to those that exist today within a single network that will work in a "network of networks" paradigm.

One part of a solution to this problem is the use of Closed User Groups. A Closed User Group is a potentially widely distributed community of users and their associated networked computer equipment who permit free and open communications within the community, but severely restrict communication to points outside the community. The use of these Closed User Groups by MAN vendors is a means of addressing the trend that network topological or geographic proximity is becoming independent of access proximity. The general concept of a Closed User Group network environment is where data packets from different enterprises never interact with each other; however all of the data packets are carried across at least part of the network on the same shared media such an OC-12 data communications link. In a MAN environment that supports closed user groups, LAN's containing host computers are identified as belonging to a specific Closed User Group, and data packets for this LAN are transported to the desired location, then validated on receipt.

To better understand this concept let's refer once again to the postal service analogy. Several postal services need to send packages to the East coast of the United States on a regular basis. At first Federal Express, United Parcel Service, and the United States Postal Services all send these packages by separate airplanes, but a bright entrepreneur offers to build a special cargo plane that will carry all three sets of packages to the East coast in a single trip. All of the services like the idea, because it saves them operating expenses, but they want assurances that the none of the packages will get mixed with packages from other postal services. The entrepreneur agrees to divide the plane into three separate cargo areas so that no mixture of packages is possible. As a result, everyone is happy and the entrepreneur now has a thriving business. The MAN vendors are very similar to this entrepreneur and the postal services can be likened to individual companies or enterprises within the MAN's coverage area. Each MAN vendor provides these separate cargo areas by assigning each enterprise to a different Closed User Group. Thus, even though data from several enterprises are traveling on the same MAN shared medium data path, the data is separated by the Closed User Group assignments.

Although the user of Closed User Groups by MAN vendors offers a partial solution to the problems of "network of networks", there are no existing solutions for managing Closed User Groups that provide protocols and management tools equivalent to those now in use within a single network. A need still exists for an improved protocol data unit (i.e., frame, cell, or packet) forwarding system which solves the above-identified problems and promotes the use of the Closed User Group paradigm, while providing a wide variety of access control tools that permit network managers to assign users to a group or groups, and then define the policy of how those groups can interact within themselves and with each other.

SUMMARY OF THE INVENTION

The present invention provides a packet processing system which contains virtual switches within physical switching systems that direct the flow of protocol data units in a data communication network. The present invention addresses the problem of providing Closed User Groups on shared medium data paths by providing protocols, algorithms, and bridge/router architectural designs that are capable of processing packets at multi-gigabyte rates while maintaining appropriate access policies and/or network security measures. By using all of these principles, the present invention reduces the cost of providing these packet switching services by enabling a single physical data switch to be divided into two or more virtual switches which individually process packets from different Closed User Groups. With reference to the postal delivery analogy, the present invention provides the details on how terminals at each airport can be designed, built, and operated to maintain separate package cargo areas for each postal service (i.e., separate virtual switches for each Closed User Group) to insure that packages from different postal services are not mixed up either before or after they are loaded onto the single airplane.

In accordance with a first aspect of the invention, a physical switching device for use in a communication network to switch OSI network layer protocol data traits within the communication network is provided. The physical switching device includes at least a first and a second virtual switch. Each virtual switch includes a decision mechanism for determining an associated directive based on a destination identifier within a particular protocol data unit received

at a data port. A processor is operatively coupled to each virtual switch to insert the particular protocol data unit into an outgoing data stream on another data port according to the associated directive to enable delivery of the protocol data unit to the destination identifier. These data ports are associated with a set of data interfaces selected from a plurality of data interfaces in a physical communication network switch. The set of data interfaces is assigned exclusively to a unique virtual switching device. These data ports can take many forms, including but not limited to, a data interface on the physical switch, a time slot out of several time slots in a time-divided frame received at a data interface on the physical switch, and a code divided cell out of several code divided cells received at one or more data interface on the physical switch.

The physical switching device preferably is designed to accommodate data interfaces of differing types such that the set of data interfaces assigned to a virtual switch may include a first data interface which manipulates a protocol data unit having a different protocol type from a second data interface such that protocol data units of different protocol types can be switched within a single virtual switch. The different protocol data unit protocol types may differ by having differing OSI physical layer media types, differing OSI link layer signaling protocols, and/or differing OSI network layer protocols.

A management apparatus is operatively coupled to each virtual switch to maintain information on an association between the plurality of data interfaces and the virtual switches. The management apparatus includes a controller dependent on the association information for limiting the processor of each virtual switch to only inserting the particular protocol data unit into an outgoing data stream on another data port associated with the same virtual switch which received the particular protocol data unit.

Further, it is desirable for the management apparatus to have a reassigning mechanism for changing a set assignment of a particular data interface such that the particular data port assignment can be moved between the virtual switching devices as needed (i.e., the data port can be moved).

Furthermore, it is necessary for the management apparatus to maintain a database of known destination identifiers and to require verification that the destination identifier in the particular protocol data unit is in the database prior to inserting the particular protocol data unit into an outgoing data stream on another data port such that delivery of the protocol data unit to an unknown destination identifier is prevented.

Each virtual switch processor preferably performs restructuring and/or monitoring operations on the particular protocol data unit. The restructuring operations include deleting, inserting, and/or replacing bits in the particular protocol data trait in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream. The monitoring operations include dropping, sending, sending a copy of, and/or auditing the contents of the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

In accordance with a second aspect of the invention, a physical switching device for use in a communication network to switch protocol data units within the communication network on a shared communication medium is provided. The physical switching device includes at least a first and a second virtual switch which is similar to that which was described in the first aspect of the present invention; however, the management apparatus is different. This different management apparatus is a virtual link management apparatus which is operatively coupled to the virtual switches. The virtual link management apparatus maintains information on at least one virtual link between at least the first and the second virtual switches. The virtual link has a first end and a second end of a data path on the shared communication medium. Each end consists of a data port from the plurality of data interfaces in the physical communication network switch.

The first and the second virtual link end of the at least one virtual link preferably are in a different set of data ports assigned exclusively to the first and the second virtual switch, respectively, such that the virtual link provides a data path between the first and the second virtual switches on the shared communication medium. The first and the second virtual switches preferably are located in a single geographic location (i.e., within the same network hardware device rack) and the shared communication medium preferably is a memory shared between the first and the second virtual switches.

Alternatively, the first and the second virtual switches may be geographically remote from one another. In addition, the first and the second virtual link ends may be in a single set of data interfaces assigned exclusively to the first and the second virtual switches such that the virtual link provides a data path between the first and the second virtual switches on the shared communication medium across a geographic distance. In this alternative arrangement, the shared communication medium preferably consists of a high data transfer rate link between the first and the second virtual switches which spans the geographic distance.

In addition, a filter may be operatively coupled to the data path which filters protocol data units communicated in either virtual link data path according to an access policy that is separately specified in each virtual switch.

In accordance with a third aspect of the invention, a communication system which delivers OSI network layer protocol data units within a first and a second virtual closed user group on a shared communication medium is provided. The communication system includes a first virtual closed user group processor for examining and modifying data bits within a protocol data unit received from a member of the first virtual closed user group on the shared communication medium. Each member of the first virtual closed user group has a unique destination identifier. The first virtual closed user group processor includes a delivery mechanism for delivering the modified protocol data unit to another member of the first virtual closed user group.

The communication system also includes a second virtual closed user group processor which is similar to the first virtual closed user group processor. The second virtual closed user group processor examines and modifies data bits within a protocol data unit received from a member of the second virtual closed user group on the shared communication medium. Also, each member of the second virtual closed user group has a unique destination identifier. In addition, the second virtual closed user group processor includes a delivery mechanism for delivering the modified protocol data unit to another member of the second virtual closed user group.

A framer is operatively coupled to the first and the second virtual closed user group processors to maintain a database of all destination identifiers currently reachable for delivery of protocol data units within the communication system. This framer preferably requires verification that each desti-

nation identifier in a protocol data unit on the shared communication medium can be currently reached for delivery through a lookup in the database, prior to completing delivery of the protocol data unit to the associated destination identifier. The framer preferably further limits access to the database such that each virtual closed user group only has access to specific destination identifiers owned by that particular virtual closed user group so that a protocol data unit having a destination identifier which is not owned by the particular virtual closed user group will not be delivered.

Each virtual closed user group processor modifies and/or monitors protocol data units. The processor modifies data bits within a received protocol data unit by deleting, inserting, and replacing bits in the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group. The processor monitors the received protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group.

In addition, each virtual closed user group processor preferably delivers the modified protocol data unit to another member of the same virtual closed user group without modifying the predetermined OSI physical layer, link layer, and network layer access protocols used to communicate protocol data units over the shared communication medium. This provides seamless integration of this closed user group functionality to LAN managers even though the LANs may be operating within a MAN as separate closed user groups. This lack of modification of the access protocols also has the advantage of enabling each virtual closed user group processor to allow any particular device capable of communicating on the shared communication medium (e.g., port with a destination identifier) to be a member of either virtual closed user group by having the framer means limit database access to destination identifiers associated with the particular device to a particular desired virtual closed user group.

The framer preferably also includes a mechanism for assigning incoming protocol data unit traffic to each virtual closed user group based on an access policy that is separately specified in each virtual closed user group.

In the preferred embodiment communication system operations of the first and the second virtual closed user group processor are performed by a first and a second virtual switch, respectively. Each virtual switch includes a decision mechanism for determining an associated directive based on a destination identifier within a particular protocol data unit received at a data port. In addition, each virtual switch includes a processor which performs the functions of the virtual closed user group delivery mechanism by inserting the particular protocol data unit into an outgoing data stream on another data port according to the associated directive to enable delivery of the protocol data unit to the destination identifier within the protocol data unit.

In an alternative embodiment to this third aspect of the present invention, the operations of the first virtual closed user group processor are divided between a first and a second virtual switch. This spreads the processing load between two virtual switches and takes into account typical communication system configurations which have many geographically separate physical switches devoted to the same closed user group.

In either embodiment, the first and the second virtual switches may be located within a single physical switching device. Both data ports for each virtual switch are then from a set of data interfaces in the physical switching device assigned exclusively to the same virtual switch.

Also in either embodiment, the first and the second virtual switches may be located within different physical switching devices. Both data ports for each virtual switch are then from a set of data interfaces in the respective physical switching devices which are assigned exclusively to the same virtual switch. As noted above, the different physical switching devices may geographically remote from one another.

Each data port may be either a data interface on a physical switching device, a time slot out of several time slots in a time-divided frame received at a data interface on a physical switching device, or a code divided cells out of several code divided cells received at at least one data interface on a physical switching device.

The physical switching device preferably is designed to accommodate data interfaces capable of manipulating different protocol types such that each set of data interfaces assigned to a virtual switch may include two or more data interfaces having mechanisms for manipulating protocol data units having different protocol types and the virtual switch is configured to switch protocol data unit coming from these data interfaces with different mechanisms. The differences in the protocol data unit data protocol types may include differing OSI physical layer media types, differing OSI link layer signaling protocols, and/or differing OSI network layer protocols.

Each virtual switch processor modifies and/or monitors protocol data units. The processor modifies data bits within a received protocol data unit by deleting, inserting, and replacing bits in the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group. The processor monitors the received protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group.

In addition, the communication system may include a virtual link between the first and the second virtual switches. This virtual link consists of a first end and a second end of a data path on the shared communication medium, where each end is a data port in a different virtual closed user group. To enforce access policies a filter may be operatively coupled to the data path to filter protocol data units communicated in the data path.

This third aspect of the invention also can be implemented in a device-implemented method to deliver protocol data units within a first and a second virtual closed user group on a shared communication medium in a communication system. This delivery method includes examining and modifying data bits within a protocol data unit received from a member of the first virtual closed user group on the shared communication medium wherein each member of the first virtual closed user group has a unique destination identifier. In addition, data bits within a protocol data unit received from a member of the second virtual closed user group on the shared communication medium are examined and modified. Each member of the second virtual closed user group also has a unique destination identifier. Further, a database of all destination identifiers currently reachable for delivery of protocol data units within the communication system is maintained. Access to this database is limited such that each virtual closed user group only has access to specific destination identifiers owned by that particular virtual closed user group. Also, verification that each destination identifier in a

protocol data unit on the shared communication medium is currently reachable for delivery through a lookup in the database is required prior to completing delivery of the protocol data unit to the associated destination identifier. Subsequently, the first virtual closed user group modified protocol data unit is delivered to another member of the first virtual closed user group after verifying that the first virtual closed user group member destination identifier is currently reachable. In addition, the second virtual closed user group modified protocol data unit is delivered to another member of the second virtual closed user group after verifying that the second virtual closed user group member destination identifier is currently reachable. This results in protocol data units having destination identifiers which are not owned by the particular virtual closed user group not being delivered anywhere.

Each examining and modifying process preferably includes modifying data bits within a received protocol data unit by deleting, inserting, and replacing bits in the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group. Alternatively, each examining and modifying process may include monitoring the received protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group.

These device-implemented steps preferably are performed such that all predetermined physical layer, link layer, and network layer access protocols used to communicate protocol data units over the shared communication medium are preserved. In other words, no changes to protocol like IP, ATM, OC-12 or the like are necessary to implement the present invention, because these steps are seamlessly integrated with this access protocols. This seamless integration enables the addition benefit that any particular device capable of communicating on the shared communication medium can be a member of either virtual closed user group by performing an additional step of adding a destination identifier associated with the particular device to the database.

These device-implemented steps preferably also include a step of assigning incoming protocol data unit traffic to each virtual closed user group based on an access policy that is separately specified in each virtual closed user group.

In addition, the device-implemented steps may include a step of providing a virtual link between the first and the second virtual closed user group. The virtual link includes a first end and a second end of a data path on the shared communication medium. Also, each virtual link end includes a data port in a different virtual closed user group. A shared memory can be used as the shared communication medium to provide the virtual link. Alternatively, a high data transfer rate link which spans a geographic distance between the first and the second virtual closed user group can be utilized to provide the virtual link. A filtering process can be performed on the virtual link such that protocol data units communicated in the virtual link data path are filtered according to an access policy.

These and various other features as well as advantages which characterize the present invention will be apparent upon reading of the following detailed description and review of the associated drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of a preferred embodiment packet processing system in accordance with the present invention.

FIG. 2 is a block diagram of a preferred embodiment physical switching device having virtual switches in accordance with the present invention.

FIG. 3 is a block diagram of an alternative preferred embodiment physical switching device which has a virtual link in accordance with the present invention.

FIG. 4 is a block diagram of another alternative preferred embodiment having virtual closed user groups in accordance with the present invention.

FIG. 5 is a block diagram providing more detail for the decision mechanism/preprocessor shown in FIG. 2 in accordance with the present invention.

FIG. 6 is a flowchart of the preferred embodiment operations of the use of virtual closed user groups as shown in FIG. 4 in accordance with the present invention.

FIG. 7 is a block diagram showing a configuration of Remote Groups in accordance with the present invention.

FIG. 8 is a block diagram showing a virtual link in accordance with the present invention.

FIGS. 9 and 10 are block diagrams showing an example of a purely segregated metropolitan network from a geographic and connectivity point of view, respectively, in accordance with the present invention.

FIGS. 11 and 12 are block diagrams showing an example of a purely segregated metropolitan network having Internet from a physical connectivity, and customer's management view, respectively, in accordance with the present invention view, respectively, in accordance with the present invention.

FIG. 13 is block diagram showing an extension of the example shown in FIGS. 11 and 12 of a purely segregated metropolitan network having Internet which also wants to provide Internetworking Service in accordance with the present invention.

FIGS. 14, 15, 16, and 17 are block diagrams showing multiple virtual networks within an enterprise from a physical, network topology, virtual routing, and consolidated router point of view, respectively, in accordance with the present invention.

FIG. 18 is a block diagram showing an example of a component hierarchy including three Virtual Routers (VRs) and multiple protocol stacks and protocol ports in accordance with the present invention.

FIG. 19 is a block diagram which shows the entire component hierarchy under the VR in accordance with the present invention.

FIG. 20 is a block diagram which shows an example of multiple cluster bridges, each associated with one bridge protocol port in accordance with the present invention.

FIG. 21 is a block diagram which shows an example of inter-VR component hierarchy which supports virtual links in accordance with the present invention.

FIG. 22 is a block diagram which shows a system overview of the VirtualRouterProcess, ForwardingAgentProcess, and the NetworkProtocolBaseProcess in accordance with the present invention.

FIG. 23 is a block diagram which shows the per port initialization required prior to packet forwarding, including Virtual Router Creation and Provisioning, in accordance with the present invention.

FIG. 24 is a block diagram which shows the per port initialization required prior to packet forwarding, including LAN Media Application Creation and Initialization, in accordance with the present invention.

FIG. 25 is a block diagram which shows the Forwarding Data Distribution—LAN Media which occurs after the LAN Media Application Creation and Initialization steps shown in FIG. 24, in accordance with the present invention.

FIG. 26 is a block diagram which shows the Protocol Binding—LAN Media which occurs after the LAN Media Application Creation and Initialization steps shown in FIG. 24, in accordance with the present invention.

FIG. 27 is a block diagram which shows the Packet Forwarding—LAN Media in accordance with the present invention.

FIG. 28 is a block diagram which shows the per port initialization required prior to packet forwarding, including Media Application Creation and Initialization—Multi-point WAN, in accordance with the present invention.

FIG. 29 is a block diagram which shows the Forwarding Data Distribution—Multi-point WAN which occurs after the Media Application Creation and Initialization steps shown in FIG. 28, in accordance with the present invention.

FIG. 30 is a block diagram which shows the Protocol Binding—Multi-point WAN which occurs after the Media Application Creation and Initialization steps shown in FIG. 28, in accordance with the present invention.

FIG. 31 is a block diagram which shows the Packet Forwarding—Multi-point WAN in accordance with the present invention.

FIG. 32 is a block diagram which shows the per port initialization required prior to packet forwarding, including Media Application Creation and Initialization—Point to Point Protocol (PPP) WAN, in accordance with the present invention.

FIG. 33 is a block diagram which shows the Forwarding Data Distribution—PPP WAN which occurs after the Media Application Creation and Initialization steps shown in FIG. 32, in accordance with the present invention.

FIG. 34 is a block diagram which shows the Protocol Binding—PPP WAN which occurs after the Media Application Creation and Initialization steps shown in FIG. 28, in accordance with the present invention.

FIG. 35 is a block diagram which shows the Packet Forwarding—PPP WAN in accordance with the present invention.

FIGS. 36 and 37 are block diagrams which show the Packet Forwarding for Virtual Link Media in accordance with the present invention.

FIG. 38 is a block diagram which shows an example where the outbound physical port is on the same logical port (LP) as the inbound physical port in accordance with the present invention.

FIGS. 39 and 40 are block diagrams which show other examples of outbound and inbound physical port assignments in accordance with the present invention.

FIG. 41 is a block diagram which shows the Packet Forwarding—Cluster Bridge Media in accordance with the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

As will be appreciated by those skilled in the art, communication networks and their operations can be described according to the Open Systems Interconnection (OSI) model which includes seven layers including an application, presentation, session, transport, network, link, and physical layer. The OSI model was developed by the International Organization for Standardization (ISO) and is described in "The Basics Book of OSI and Network Management" by Motorola Codex from Addison-Wesley Publishing Company, Inc., 1993 (First Printing September 1992).

Each layer of the OSI model performs a specific data communications task, a service to and for the layer that precedes it (e.g., the network layer provides a service for the transport layer). The process can be likened to placing a letter in a series of envelopes before it's sent through the postal system. Each succeeding envelope adds another layer of processing or overhead information necessary to process the transaction. Together, all the envelopes help make sure the letter gets to the right address and that the message received is identical to the message sent. Once the entire package is received at its destination, the envelopes are opened one by one until the letter itself emerges exactly as written.

In a data communication transaction, however, each end user is unaware of the envelopes, which perform their functions transparently. For example, an automatic back teller transaction can be tracked through the multilayer OSI system. One multiple layer system (Open System A) provides an application layer that is an interface to a person attempting a transaction, while the other multiple layer system (Open System B) provides an application layer that interfaces with applications software in a bank's host computer. The corresponding layers in Open Systems A and B are called peer layers and communicate through peer protocols. These peer protocols provide communication support for a user's application, performing transaction related tasks such as debiting an account, dispensing currency, or crediting an account.

Actual data flow between the two open systems (Open System A and Open System B), however, is from top to bottom in one open system (Open bottom to top in the other open system (Open System B, the destination). Each time that user application data passes downward from one layer to the next layer in the same system more processing information is added. When that information is removed and processed by the peer layer in the other system, it causes various tasks (error correction, flow control, etc.) to be performed. The user is unaware of any of this, of course, but in fact that's what's happening while the words, "Please wait, your transaction is being processed" appears on the screen.

The ISO has specifically defined all seven layers, which are summarized below in the order in which the data actually flow as they leave the source:

- Layer 7, the application layer, provides for a user application (such as getting money from an automatic bank teller machine) to interface with the OSI application layer. That OSI application layer has a corresponding peer layer in the other open system, the bank's host computer.
- Layer 6, the presentation layer, makes sure the user information (a request for $50 in cash to be debited from your checking account) is in a format (i.e., syntax or sequence of ones and zeros) the destination open system can understand.
- Layer 5, the session layer, provides synchronization control of data between the open systems (i.e., makes sure the bit configurations that pass through layer 5 at the source are the same as those that pass through layer 5 at the destination).
- Layer 4, the transport layer, ensures that an end-to-end connection has been established between the two open

systems and is often reliable (i.e., layer 4 at the destination "confirms the request for a connection," so to speak, that it has received from layer 4 at the source).

Layer 3, the network layer, provides routing and relaying of data through the network (among other things, at layer 3 on the outbound side an "address" gets slapped on the "envelope" which is then read by layer 3 at the destination).

Layer 2, the data link layer, includes flow control of data as messages pass down through this layer in one open system and up through the peer layer in the other open system.

Layer 1, the physical interface layer, includes the ways in which data communications equipment is connected mechanically and electrically, and the means by which the data move across those physical connections from layer 1 at the source to layer 1 at the destination.

The first and primary item in the Virtual Routing (VR) toolbox is the notion of a Virtual Router (i.e., virtual switch). Traditionally, multiprotocol bridge/routers have a single version of IP, bridging, IPX, etc. in operation, which coordinates the flow of traffic between all ports activated for the protocol. Single events on one incoming port can simply or profoundly affect the flow of traffic on other ports.

To implement virtual routers, object oriented software techniques are used to create separate instances of multiprotocol bridge/router code residing on the same router platform. Each operates independently of one another and is not directly aware of the existence of other virtual routers.

The simplest example of Virtual Routing consists of the case where the population of physical interfaces **102**, **110** on the router **150** is partitioned among the Virtual Routers **152**, **154**, **156** running in the chassis as shown in FIG. 1. This is a software partitioning, which means that:

All management continues to be done through a single, possibly redundant control processor.

The partitioning of connectivity into Virtual Routers **152**, **154**, **156** is done on a per interface basis, rather than a per card basis. As we'll see later, the partitioning can be further refined to individual media destinations on selected media.

What this means is that, in a first step, an owner of the multiprotocol bridge/router **150** has now been given three routers **152**, **154**, **156** for the price of one. If they are servicing the needs of several organizations, the cost of a highly available and manageable enterprise router can be spread over several independent clients.

The cost savings in routers can be significant. However, over time the cost of wide area network services can dwarf the up front router equipment costs. The second step in Virtual Routing is to permit the media **158** to be shared in such a way as to give each Closed User Group the impression that they have a medium to themselves.

A solution can be provided for this problem through the use of remote groups. Remote groups are a means of sharing multipoint wide area networks among several Closed User Groups. This is done rather simply:

Multi-point WAN's such as Frame Relay, X.25 and ATM work with the concept that a media address is prepended to the message payload. This media header is used by the network to determine how the message shall be delivered. Switched networks such as ATM often have a prerequisite signaling process which determines the header to be used to reach a certain destination.

A multipoint WAN interface will remain informed of all the destinations which are currently reachable through the WAN. The WAN internals will refer to them through the media address in the header. The interface is free to send or receive messages from any of these active remote destinations.

The WAN software may be split into two portions: A Framer to maintain the state of the WAN connections, and a set of Remote Groups which are responsible for examining and processing the contents of the message. The Remote Group software more or less believes that it is the exclusive owner of the WAN interface; however the Framer only gives the Remote Group information on the specific media addresses owned by that group.

One of the virtues of Remote Groups is that it does not modify the protocols used to communicate over the shared medium in any way. Specifically, any device capable of talking to the medium can be a member of a Remote Group.

The result of such a configuration of Remote Groups is shown in FIG. 7. A single multipoint WAN has been carved into several multipoint WAN's, each of which may be used to service the needs of a different community. The example Frame Relay network has been split into two independently operating networks. It will be appreciated by those skilled in the art that multiple telecommunication interfaces and local loops to the Frame Relay service could also have provided this connectivity, but at considerably greater expense (i.e., more hardware).

Most significantly, a single multipoint WAN has been taken and divided such that a subset of it is available to each of several Virtual Routers. The administrative isolation of Virtual Routers have now been extended from within a chassis to independently operating systems of routers sharing both common chassis and a media backbone.

So far the discussion has focused on techniques where groups can be entirely sealed from each other. However, there are reasons why these barriers should be partially torn down in the interests of better communications. Examples include:

Permitting access for certain protocol families. For example, Appletalk and IPX are not noted for scaling well over large domains, while IP does a good job of this. Virtual Routing permits the limited area protocols to run fully separately, while the two IP regions are interconnected.

Permitting certain applications to run. For example, many sites are comfortable receiving IP electronic mail traffic from any point, do not wish general access. If traffic moves between domains at a small number of constrained points it is easy to exercise policy at only those points rather than throughout the network.

More than anything else, when moving from one user group to another, a change in access policy can occur. Every group wants to define their own policy for accepting traffic each other group. Thus, a mechanism is needed not only to pass traffic between user groups, but also a means whereby the owners of each user group can define the filtering desired for packets leaving and entering the group.

As shown in FIG. 8, through the use of a networking medium Virtual Routers, which may be real, separate routers, can be connected. Within a chassis containing several Virtual Routers, a point to point link between two of these Virtual Routers can be defined which is a complete software artifact—each Virtual Router defines an interface (i.e., sometimes called a Protocol Port) that is one end of this Virtual Link "pipe". Messages inserted in one end of the pipe logically arrive at an incoming port of the other Virtual Router.

In addition to permitting the raw movement of traffic, Virtual Links permit protocol filters for traffic moving to and from the link. In fact, the filters on these links are often the most important in the design of a network, because a change from one network of routers and Virtual Routers to another coincides with a change in administrative policy as traffic moves from one domain to another.

Virtual Links are seen as point to point links by all networking protocols which reside in the unit. Thus, the Virtual Link can be used to carry IP, IPX, Appletalk and other internetworking traffic. Routing updates from protocols such as DECnet Routing will flow over the Virtual Link to update the routing tables on the other virtual router. This is the only way that the protocols in the separate Virtual Routers can communicate with each other, despite the fact that they run in the set of physical processors.

Forwarding data through a Virtual Link performs all the activity associated with an extra "hop" between two networks. Time to Live counts are decremented and all filters are applied as if it were a real link. However, since a Virtual Link is a software artifact the overhead is small; the forwarding algorithms are applied to the packet in succession when it arrives from an external medium, and it is not moved anywhere until an ultimate destination outside the box is found. Routes that involve packets progressing through multiple Virtual Links within the chassis are both possible and reasonable in complex configurations.

So far a service provider has been permitted to segregate traffic between customers but it is also necessary to manage this segregation. Since "Network Management" means too many different things to different people, a few terms should be defined for this discussion.

- Surveillance consists of the act of monitoring the status of network entities. Modifications of network parameters must take place through other channels.
- Provisioning consists of the act of modifying the configuration or operation of network equipment, either temporarily or permanently.
- Data Privacy in this context means that the client has high assurance that other parties (other than the service provider) cannot intercept the user's data packets. Network Privacy means that the details of the customer's network operations are not available to other customers.
- Security in this management context means that no act on the part of another customer, however malicious, will affect operations of the closed user group.

Also, two different techniques are available to survey and provision network equipment:

- Simple network management protocol (SNMP), the Internet standard which has an assured place as the de facto technique for the management of all network equipment. SNMP is noted as being pretty effective for surveillance and light duty provisioning. Its limitations are that it cannot survey items specific to a vendor (such as chassis status) or perform massive provisioning (such as initial setup of the unit) without resort to vendor extensions.
- CAS (Component Administration System). The internal network surveillance and provisioning system specific to Network Systems Corporation equipment.

Both have their complementary attributes; SNMP is an excellent platform for monitoring the health and status of a remote, network attached device. With the advent of SNMP Version 2, it has become suitable for "tweaking" devices through setting external parameters. A direct dialogue with CAS is better suited for wholesale changes in the configuration of the unit; CAS also permits access to the internal control features of a particular piece of network equipment which are not defined in the standard SNMP vocabulary.

To best take advantage of these complementary features, the following should be done:

- Access through CAS has an omniscient view of the box. All components are available for inspection or alteration by properly authorized CAS users. Once a CAS user has read-only or modification rights on the chassis, all parameters of all virtual routers are available.
- SNMP users work with a copy of SNMP which is part of their Virtual Router. Proper authorization permits one of two alternative views of the chassis. In "omniscient" mode, they perceive all interfaces in the unit and can modify their parameters. In "local" mode all interfaces are given an interface number for SNMP purposes, but the ones not owned by the virtual router will appear stubbornly offline regardless of their actual status. This permits the actual owner of the router access to the entire interface population, while Virtual Router clients have access to only their own internal interfaces.
- SNMP based "core" configuration information not associated with interfaces, such as filtering, IP routing parameters, DECnet node addresses, etc. are only accessible from a host which can reach that Virtual Router. Since SNMP management requires IP, access to this core information requires a Virtual Link or other mechanism giving access to that Virtual Router.
- Under no circumstances can an SNMP user modify the router variables for a virtual router other than their own. If such a facility is provided, it can either be done through CAS or by providing access to the SNMP stack of a "distant" Virtual Router through a Virtual Link.

To better understand these principles, the following examples are given.

A purely segregated metropolitan network, shown in FIGS. **9** and **10** will be the simplest example of the set, because the situation is simplest. A utility oriented company, Lightco, happens to own a large fiber optic cable plant that can be used to access local businesses. They choose to offer LAN interconnection services to these businesses as an additional revenue opportunity for themselves.

Because it is an existing proven technology and well suited for the cable plant, FDDI is chosen as the backbone medium. In the example, three enterprises are to be connected to a common FDDI metropolitan area network, with Points of Presence required at three different sites.

The routers at a site serving one client run a single Virtual Router (or one Real Router, if you like). The ones serving two clients run one Virtual Router for each client.

The advantages of Virtual Routing come into more into play when routing (e.g., incorporating public network access) becomes part of the picture. In the hypothetical example, connectivity to the Internet is offered through a drop at the Point of Presence (POP) of an Internet service provider. A Corp. wants unrestricted Internet access (or, more accurately, they will take responsibility for access within their own network). C Corp. wants mail access to a single machine on their network, and B Corp. is not interested in Internet access at all. How can these divergent needs be handled?

One solution is shown in FIGS. **11** and **12**. Let us look at the noteworthy items in this new configuration:

- The FDDI backbone has been replaced with an ATM backbone. The customer is oblivious to the change

(other than by examining the unit internals through network management). For simplicity, Permanent Virtual Circuits (PVC's) are employed in this metropolitan ATM network. One PVC must be established between every pair of virtual routers that wish to communicate. For example, chassis **224** will require four PVC's. One to the chassis' **224** B Corp. virtual router to the chassis **226**, one from chassis **224** A Corp. virtual router to the chassis **230**, and one each from the each chassis **224** virtual router to each customer's Virtual Router on the chassis **232**.

Each Virtual Router in the chassis **232** is probably administered by the Internet service provider, where they offer the customer the right to inspect the current statistics of that router via SNMP. That Virtual Router takes traffic off the ATM network and performs IP routing on it. That Virtual Router has a single other port with an IP address indicating it belongs on the distribution LAN at the Internet Service provider's POP.

Each Virtual Router serves as a mechanism to filter packets according to the customer's expectations.

For A Corp., filters are installed which permit unlimited access to one specific IP address, and default filters deny Telnet, rlogin, FTP, etc. access from the Internet to all other hosts.

For the more paranoid C Corp., all IP packets directed from the Internet to all hosts but one are denied. On that single host, packets for Simple Mail Transfer Protocol (SMTP) Domain Name Service, and Internet Control Message Protocol (ICMP) are permitted to pass.

Again taking the physical examples above, the technologically progressive A Corp. chooses to switch to a routed backbone as shown in FIG. 13, rather than the bridged backbone in the previous examples. For simplicity, it is assumed that they want to route IP while continuing to bridge "other" traffic. Very little needs to change from the previous example. Steps have already been taken as needed to segregate A Corp.'s traffic from all others, so all that is needed is to concentrate on A Corp.'s concerns within their virtual routers. Thus, if we look at A Corp.'s virtual network in isolation, perhaps giving the situation shown in FIG. **13**. The ATM PVC's and bridging parameters stay the same—only IP is activated for the ports on and off the ATM MAN. Open Shortest Path First (OSPF) or Routing Information Protocol (RIP) is run in each of the virtual routers so that automatic route discovery may take place.

This means that the entire idea is that things become simple at this level once virtual routing is in place. It should be noted what has been done in IP terms:

The MAN carrier has isolated the proper portion of their connectivity plant and given it to the customer. The customer is free to assign a subnet that they own to this segment, permitting a routed IP network to be used to interconnect the many subnets which presumably lie at each site. This feat can be duplicated for any desired number of routing customers.

The three routers are SNMP visible to the customer. Given proper authority, they may alter parameters which are settable via SNMP. Since they are isolated to their Virtual Routers the privacy and integrity of other customer's data is unaffected.

Routers supporting traffic for several customers must be routers with Virtual Networking. Routers to service a single customer can be of any make offering suitable connectivity.

The ATM backbone is visible to the customer as an ATM network. If the MAN carrier has a more complex backbone than that, FDDI can be disguised such that the internals of the MAN plant from the customer and have it appear as a "cloud". Native ATM transfers may take place from the same router to service the needs of different customers.

Finally, it should be noted that this more sophisticated customer is using routers of their own as a gateway between their numerous LAN's and the metropolitan access node. A metropolitan service provider which feels up to handling the administrative work might consider offering multiple LAN connections to the user at each site, where traffic is routed between the LAN's at little incremental expense to the client. Since all these LAN's belong to the same Virtual Router, the generally unlimited access policies that exist within a site can be followed while continuing to prohibit access to other clients, even if they share the same physical router. Pricing of such a service has to be aimed so that it is cost effective for the client to not purchase and administer their own router.

So far several cases have been covered where there is an easily well defined vendor and customer relationship between the administrators and users of virtual routing. However, similar situations arise in single organizations where the constituents have differing needs and priorities. This is also known as enterprise networking with divisional autonomy and is shown in FIGS. **14**, **15**, **16**, and **17** from a physical, network topology, virtual routing, and consolidated router point of view, respectively. For example:

Corporations or government agencies with a highly decentralized structure, where each division really wanted its "own" network.

Regional networking cooperatives which maintain a Wide Area Network to be shared by its constituent members.

Let us look at a Corporate problem to see how Virtual Routing techniques may address these issues. For the last example, the case of the recently merged Alpha and Beta divisions of MegaCorp having come to cohabit the same campus. The networking group and the expensive wide area network are run by MegaCorp corporate, but the fiercely independent divisions wish to be isolated from each other. They offer reasons a management consultant might find both good and bad for wanting to do so:

Both have retained their own computer support people who install hosts, servers, and wiring hubs in their respective departments. These people and their managers have gracelessly conceded to a common WAN and campus backbone, but staunchly resist attempts to control how "their" hosts will talk within the division.

For historical reasons, both have different IP network numbers assigned to them.

The Alpha group once paid for a fiber optic connection between two buildings which they will use in preference to the (slower) company routed backbone. They want to use the backbone as backup for their private link, however.

Both run small pockets of Netware (IPX) applications scattered throughout their empires. Hosts have been known to inadvertently access the other division's servers, which was patched up through an elaborate, fragile series of bridge filters.

Both are large Appletalk users, but have no reason to permit Appletalk interconnection.

E-mail is commonly sent between both divisions and Corporate, and there are IP based servers in each which are used to archive information the other division might want or need.

Both want Internet access through Corporate. They will determine access policy to the Internet within their division.

To give the problem a tangible feel and a sense of reality, an illustration of the physical systems is shown in FIG. **14**.

There are number of ways to solve these problems. The simplest is the "brute force" approach, where in fact separate network plants are provided for each of our client customers and explicit, physical connections are built between them. One example is shown in FIG. **14**. Some of the virtues of this configuration include:

The Appletalk and IPX problem is solved neatly. IP is routed and a Mac layer bridging is used for everything else. Turning on Appletalk or IPX routing in each separate network will present no difficulties if either group independently decides they would prefer to route these protocols.

The fiber link remains the property of Alpha, while the Frame Relay network will continue to work in a pinch.

The vexing IP subnetting rule that you must stay within a subnet to reach any point on it is eliminated. There are entirely separate Frame Relay nets, each with a subnet owned by the separate division.

IP filtering between the two networks is easy to administer. All data moving between the two network arrive at the Corporate router, who halts non-IP traffic and administers policy on the combination of Internet and other division traffic that will enter each divisional domain.

So why not do it this way? To do it the "brute force" way with conventional bridge routers, 5 Frame Relay local loops are needed instead of 3 and 6 Backbone and corporate routers are needed instead of 3.

With a lot of tinkering on "classic" bridge/routers, the ingenious corporate network planner might be able to accommodate these people with a single backbone and some filtering. However any changes will have the habit of bringing the entire house of cards down, causing long service times on the core routers and frequent complaints following maintenance. Clearly a more controllable scheme is desired. A solution involving the construction of a set of filters on a "Brand X" bridge/router which meets the user requirements should be readily understood by those skilled in the art so it will not be provided here.

Using Virtual Routers, a solution to this complex problem becomes more elegant. If the "brute force" configuration is referred to again and viewed from a network topology view as shown in FIG. **15** and reconfigure it in a Virtual Routing environment where the backbone routers are physical routers capable of virtual routing. The result is shown in FIG. **16**. This particular "brute force" configuration was deliberately chosen among several brute force options to more easily illustrate the present invention; however, one of ordinary skill in the art will appreciate that these principles can be applied to any "brute force" solution provided they are properly reconfigured into a virtual routing scheme.

Several interesting things are apparent about this configuration, including:

The brute force configuration had **6** backbone routers; now only one at each physical site is used.

There is one Frame Relay connection and local loop per physical site.

The FDDI ring at Corporate has been changed so that traffic for both Alpha and Beta can flow over it.

Connectivity between the different divisions takes place through Virtual Links within the backbone router at Corporate. Filters on the Virtual Links are used to enforce access policy.

The Corporate network lies within a Virtual Router of its own, for administrative convenience.

Even the network shown in FIG. **16** contains more equipment than is strictly necessary. Note that multiple routers are still present at each site; one for the "backbone" and one which is owned by the maintenance personnel of each division. If these disparate network management groups become sufficiently trusting (or consolidated) that they can be persuaded simply to leave other Virtual Routers alone when configuring their network, then the configuration can be reduced to the consolidated block diagram shown in FIG. **17**. To reach this configuration, the redundant routers have been eliminated and the Corporate FDDI ring has been replaced with a set of fiber optic repeaters.

Now different groups share the same router and exercise their different concerns. Is such mutual trust warranted? Again, this brings up the difference between security and integrity—integrity against unintended alterations since each group is dealing with an independent set of IP protocol stacks, management software, and interfaces, and they have no excuse to tinker with those belonging to others. Their interests only conflict when modifying the physical health of the common Frame Relay network they share. However, security is not provided in that one group is not immune from the malicious intent of the other.

Hopefully, these example have shown that there is a new networking problem beginning to emerge-the problem of networking networks, rather than networking hosts as in the past. Just as peer to peer host networks required a different model of what networks were about at the time if its introduction, the interconnection of networks also requires new tools and approaches to be properly implemented.

With the tools discussed above and further discussed below in detail, the present invention provides solutions for this new problem. Additional tools will be required in the future to accommodate exciting new technologies such as wireless, public networks for mobile computing; however, with the proper foundations provided by the present invention it should be readily understood how to deal with such new technologies as they are developed and implemented in the data networking environment.

Referring to once again to FIG. **1**, a preferred embodiment of the present invention which provides a packet processing system **150** which contains virtual switches **152**, **154**, and **156** within physical switching systems **150** that direct the flow of protocol data units into inbound interfaces **102** and out of outbound interfaces **110** in a data communication network is shown. This is similar to the partitioning of a large shared network hard disk into several disk partitions and restricting access by different users to different partition. For example, a 750 Megabyte hard drive can be partitioned into three 250 Megabyte partitions. Further, each partition can be password protected so that only users which know the correct password have access to that hard disk partition. All members of a Closed User Group would know the correct password for the partition assigned to that Closed User Group and no one outside of that group would know the correct password.

In a similar manner, the present invention uses Closed User Groups to provide access to a shared medium data path **158**. This access is accomplished by providing protocols, algorithms, and bridge/router architectural designs that are capable of processing packets at multi-gigabyte rates while maintaining appropriate access policies and/or network security measures. By using all of these principles, the

present invention reduces the cost of providing these packet switching services by enabling a single physical data switch **150** to be divided into two or more virtual switches **152**, **154**, and **156** which individually process packets from different Closed User Groups. With reference to the hard disk partitioning analogy, the following present invention detailed description provides a set of operating techniques, device architectures, and constraints necessary for partitioning a communication network switch, like a hard disk, so that different Closed User Groups can have access to different partitions (i.e., virtual switching devices) while ensuring that access to the different partitions is limited to members of the Closed User Groups.

In a preferred embodiment of the present invention shown in FIG. **2**, a physical switching device **150** for use in a communication network to switch OSI network layer protocol data units within the communication network is provided. The physical switching device **150** includes at least a first **152** and a second **154** virtual switch. Each virtual switch **152**, **154** includes a decision mechanism **104** for determining an associated directive based on a destination identifier within a particular protocol data unit **140** received at a data port **160**. A processor **108** is operatively coupled to each virtual switch **152**, **154** to insert the particular protocol data unit **140** into an outgoing data stream on another data port **162** according to the associated directive to enable delivery of the protocol data unit **140** to the destination identifier. These data ports **160**, **162** are associated with a set of data interfaces **112**, **114**, **116**, **118**, **132**, **134**, **136**, and **138** selected from a plurality of data interfaces in a physical communication network switch **150**. The set of data interfaces **112**, **114**, **116**, **118**, **132**, **134**, **136**, and **138** is assigned exclusively to a unique virtual switching device **152**. These data ports **160**, **162** can take many forms, including but not limited to, a data interface **112** on the physical switch **150**, a time slot out of several time slots in a time-divided frame received at a data interface **112** (e.g., an FDDI time multiplexed optical fiber) on the physical switch **150**, and a code divided cell out of several code divided cells received at one or more data interface **112** (e.g., in ATM cells may be sent over several different paths and reassembled in a sequence based on a cell identifier within the cell header information) on the physical switch **150**. In addition, two or more data ports **195** and **197** can be associated with one or more data interfaces and retrieve protocol data units from the data interface data stream for each data port **195** and **197** based on unique attributes (e.g., destination identifiers) associated with a particular data port.

The physical switching device **150** preferably is designed to accommodate data interfaces of differing types such that the set of data interfaces assigned to a virtual switch **152** may include a first data interface **114** which manipulates a protocol data unit having a different protocol type from a second data interface **116** such that protocol data units of different protocol types can be switched within a single virtual switch **152**. The different protocol data unit protocol types may differ by having differing OSI physical layer media types, differing OSI link layer signaling protocols, and/or differing OSI network layer protocols.

A management apparatus **164** is operatively coupled to each virtual switch **152**, **154** to maintain information on an association between the plurality of data interfaces and the virtual switches. The management apparatus **164** includes a controller **166** dependent on the association information for limiting the processor **108** of each virtual switch **152**, **154** to only inserting the particular protocol data unit **140** into an outgoing data stream on another data port **162** associated with the same virtual switch **152** which received the particular protocol data unit **140**.

Further, it is desirable for the management apparatus **164** to have a reassigning mechanism **168** for changing a set assignment of particular data interface **118** such that the particular data interface assignment **170** can be moved between the virtual switching devices **152** and **154** as needed (i.e., the data port **160** can be moved).

Furthermore, it is necessary for the management apparatus **164** to maintain a database **172** of known destination identifiers and to require verification that the destination identifier in the particular protocol data unit **140** is in the database **172** prior to inserting the particular protocol data unit **140** into an outgoing data stream on another data port **162** such that delivery of the protocol data unit **140** to an unknown destination identifier is prevented.

Each virtual switch processor **108** preferably performs restructuring and/or monitoring operations on the particular protocol data unit **140**. The restructuring operations include deleting, inserting, and/or replacing bits in the particular protocol data unit **140** in accordance with the associated directive prior to inserting the particular protocol data unit **140** into the outgoing data stream. The monitoring operations include dropping, sending, sending a copy of, and/or auditing the contents of the particular protocol data unit **140** in accordance with the associated directive prior to inserting the particular protocol data unit **140** into the outgoing data stream.

In accordance with an alternative embodiment of the present invention shown in FIG. **3**, a physical switching device **150** for use in a communication network to switch protocol data units within the communication network on a shared communication medium is provided. The physical switching device **150** includes at least a first **152** and a second virtual switch **152** which is similar to that which was described in the preferred embodiment of the present invention; however, the management apparatus is different. This different management apparatus is a virtual link management apparatus **174** which is operatively coupled to the virtual switches **152**, **154**. The virtual link management apparatus **174** maintains information on at least one virtual link **176** between at least the first **152** and the second **154** virtual switches. The virtual link **176** has a first end **178** and a second end **180** of a data path **182** on the shared communication medium **158**.

The virtual link **176** provides a data path **182** between the first **152** and the second **154** virtual switches on the shared communication medium **158**. The first **152** and the second **154** virtual switches preferably are located in a single geographic location (i.e., within the same network hardware device rack) and the shared communication medium **158** preferably is a memory shared between the first **152** and the second **154** virtual switches.

Alternatively, the first **152** and the second **154** virtual switches may be geographically remote from one another. In addition, the first **178** and the second **180** virtual link ends may be in a single set of data ports assigned exclusively to the first **152** and the second **154** virtual switches such that the virtual link **176** provides a data path **182** between the first **152** and the second **154** virtual switches on the shared communication medium **158** across a geographic distance. In this alternative arrangement, the shared communication medium **158** preferably consists of a high data transfer rate link between the first **152** and the second **154** virtual switches which spans the geographic distance.

In addition, filters **184** and **185** (associated with virtual switch **152** and **154**, respectively) may be operatively

coupled to the data path **182** which filters protocol data units communicated in the virtual link data path **182**. Each filter **184** or **185** performs filtering operations according to one access policy out of a plurality of access policies that are separately specified for each virtual switch.

The present invention may also be described in reference to yet another preferred embodiment shown in FIGS. **2** and **4**. Referring to FIG. **4**, a communication system **190** which delivers OSI network layer protocol data units within a first **186** and a second **188** virtual closed user group on a shared communication medium **158** is provided. It will be noted that the shared communication medium **158** may include a variety of physical communication media (as shown in FIG. **4**), in addition to other intermediate network bridges, routers and the like. The communication system **190** includes a first virtual closed user group processor **152** for examining and modifying data bits within a protocol data unit received from a member of the first virtual closed user group **186** on the shared communication medium **158**. Each member of the first virtual closed user group **186** has a unique destination identifier. As shown in FIG. **2**, the first virtual closed user group processor **152** includes a delivery mechanism **108** for delivering the modified protocol data unit **140'** to another member of the first virtual closed user group **186**.

As shown in FIG. **4**, the communication system **190** also includes a second virtual closed user group processor **154** which is similar to the first virtual closed user group processor **152**. The second virtual closed user group processor **154** examines and modifies data bits within a protocol data unit received from a member of the second virtual closed user group **188** on the shared communication medium **158**. Also, each member of the second virtual closed user group **188** has a unique destination identifier. In addition, as shown in FIG. **4**, the second virtual closed user group processor **154** includes a delivery mechanism **108'** for delivering the modified protocol data unit **141'** to another member of the second virtual closed user group **188**.

A framer **164** is operatively coupled to the first **152** and the second **154** virtual closed user group processors to maintain a database **192** of all destination identifiers currently reachable for delivery of protocol data units within the communication system **190**. This framer **164** preferably requires verification that each destination identifier in a protocol data unit on the shared communication medium **158** can be currently reached for delivery through a lookup in the database **192**, prior to completing delivery of the protocol data unit to the associated destination identifier. The framer **164** preferably further limits access to the database **192** such that each virtual closed user group **186**, **188** only has access to specific destination identifiers owned by that particular virtual closed user group **186** or **188** so that a protocol data unit having a destination identifier which is not owned by the particular virtual closed user group **186** or **188** will not be delivered.

Each virtual closed user group processor **152**, **154** modifies and/or monitors protocol data units. The processor **152** modifies data bits within a received protocol data unit by deleting, inserting, and replacing bits in the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group. The processor **152** monitors the received protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group.

In addition, each virtual closed user group processor **152**, **154** may deliver the modified protocol data unit to another destination within the same virtual closed user group without modifying the predetermined OSI physical layer, link layer, and network layer access protocols used to communicate protocol data units over the shared communication medium. This provides seamless integration of this closed user group functionality to LAN managers even though the LANs may be operating within a MAN as separate closed user groups. This lack of modification of the access protocols also has the advantage of enabling each virtual closed user group processor **152**, **154** to allow any particular device capable of communicating on the shared communication medium (e.g., port with a destination identifier) to be a member of either virtual closed user group by having the framer **164** limit database **192** access to destination identifiers associated with a particular destination within a particular desired virtual closed user group.

The framer **164** preferably also includes a mechanism for assigning incoming protocol data unit traffic to each virtual closed user group **186**, **188** based on an access policy that is separately specified in each virtual closed user group.

In the preferred embodiment communication system **190** operations of the first **152** and the second **154** virtual closed user group processor are performed by a first **152** and a second **154** virtual switch, respectively. Each virtual switch **152**, **154** includes a decision mechanism **104**, **104'** for determining an associated directive based on a destination identifier within a particular protocol data unit **140**, **141** received at a data port **160**, **194**. In addition, each virtual switch **152**, **154** includes a processor **108**, **108'** which performs the functions of the virtual closed user group delivery mechanism by inserting the particular protocol data unit **140**, **140'** into an outgoing data stream on another data port **162** or **196** according to the associated directive to enable delivery of the protocol data unit **140** or **141** to the destination identifier within the protocol data unit.

In an alternative embodiment to this preferred embodiment described in reference to FIGS. **2** and **4**, the operations of the first virtual closed user group processor **152** are divided between a first and a second virtual switch. This spreads the processing load between two virtual switches and takes into account typical communication system configurations which have many geographically separate physical switches **150**, **150'** devoted to the same closed user group (e.g., group **186**).

In either embodiment, the first **152** and the second **154** virtual switch may be located within a single physical switching device **150**. These data ports **160** and **162** as well as **194** and **196** for each virtual switch are then from a set of data interfaces in the physical switching device **150** assigned exclusively to the same virtual switch **152** and **154**, respectively.

Also in either embodiment, the first **152** and the second **154** virtual switches may be located within different physical switching devices **150** and **150'**. These data ports **160** and **162** as well as **194'** and **196'** (data ports **194'** and **196'** for virtual switch **154'** are not shown, but are the same as those shown in FIG. **2**) for each virtual switch are then from a set of data interfaces in the respective physical switching devices **150** and **150'** which are assigned exclusively to the same virtual switch **152** and **154'**. As noted above, the different physical switching devices **150** and **150'** may be geographically remote from one another.

As previously noted, these data ports **160**, **162**, **194**, and **196** may be either a data interface **112** on a physical switching device **150**, a time slot out of several time slots in a time-divided frame received at a data interface **112** on a physical switching device **150**, or a code divided cells out of

several code divided cells received at at least one data interface **112** on a physical switching device **150**. In addition, two or more data ports **195** and **197** can be associated with one or more data interfaces and retrieve protocol data units from the data interface data stream for each data port **195** and **197** based on unique attributes (e.g., destination identifiers) associated with a particular data port.

In addition, as previously noted, the physical switching device **150** preferably is designed to accommodate data interfaces capable of manipulating different protocol types such that each set of data interfaces assigned to a virtual switch **152** may include two or more data interfaces **114, 116** having mechanisms for manipulating protocol data units having different protocol types and the virtual switch **152** is configured to switch protocol data unit coming from these data interfaces **114, 116** with different mechanisms. The differences in the protocol data unit data protocol types may include differing OSI physical layer media types, differing OSI link layer signaling protocols, and/or differing OSI network layer protocols.

Also, as previously noted, Each virtual switch processor **108** modifies and/or monitors protocol data units.

In addition, the communication system **190** may include a virtual link **76** between the first **152** and the second **154** virtual switches. This virtual link **176** consists of a first end **178** and a second end **180** of a data path **182** on the shared communication medium **158**, where each end **178, 180** is a data port in a different virtual closed user group **186, 188**. To enforce access policies a pair of filters **184** and **185** may be operatively coupled to the data path **182** to filter protocol data units communicated in the data path **182** based on individual access policies that are separately specified for each virtual closed user group.

The present invention virtual switching apparatus and method are integrated into a particular communication network device that forwards packets. The following discussion details this integration; however many of the forwarding operations are more thoroughly discussed in the previously identified related U.S. patent application Ser. No. 08/366, 221 entitled "Method And Apparatus For Accelerated Packet Forwarding".

Referring to FIGS. **2** and **5**, a preferred embodiment of a forwarding system **100** in which a protocol data unit preprocessor **104** (also termed a fast packet processor (FPP)) is used in a protocol data unit forwarding device **108** that operates in a communication network to transfer protocol data units (e.g., **140**) within the communication network. The forwarding device **108** manipulates bits of information at the OSI network, link and physical layers and preferably performs as one or more network devices including, but not limited to, a bridge, a router, a switch, a line filter, a protocol converter, an encapsulating device, and a security device. It will be appreciated that various types of communication networks exist which utilize forwarding devices that perform these functions including local protocol data unit source devices (e.g., desktop computers or workstations), local area networks, wide area networks, metropolitan area networks, and wireless networks. Also, it will be appreciated by those skilled in the art that the forwarding device **108** may perform other network-based functions without departing from the scope and spirit of the present invention. In addition, other types of data in the communication network could readily be manipulated by the forwarding device **108**.

The forwarding device **108** includes an inbound interface **102** and outbound interface **110** which control the flow of protocol data units **140** and **140'** into and out of the forwarding device **108**, respectively. These interfaces **102** and **110** are configured differently depending on the type of communication network that the forwarding device **108** is connected to as well as the particular location within such a network that the forwarding device **108** is located.

Turning now more specifically to FIG. **5**, an example of how the decision mechanism **104** may be implementing within the forwarding device **108** is shown. The preprocessor **104** (i.e,. one possible form of decision mechanism **104**) includes an identifier **122** which determines media header information of a protocol data unit received from over the communication network. The identifier **122** preferably analyzes the inbound stream of data bits from the inbound interface **102** to find media header information of a protocol data unit **140** received from over the communication network. In addition, a validation mechanism **124** is operatively coupled to the identifier **122** to validate the media header information. Also, a modifier device **126** is operatively coupled to the identifier **122** to add next operation information in the form of an associated directive **142** to the media header information based upon the determined media header information such that subsequent processing of the protocol data unit **140** by a protocol data unit forwarding processor **108** is reduced. This next operation information preferably includes bits necessary to enable the use of two or more virtual switches inside a single physical switching device. The modified media header information including the associated directive **142** along with the remaining portion of this particular protocol data unit **140** are then stored in a memory buffer **106** until the forwarding processor **108** is able to process this particular protocol data unit. The forwarding processor **108** is operatively coupled to the preprocessor **104** and the memory buffer **106** to forward the protocol data unit **140'** without the modified media header information in the communication network based upon the next operation information contained in the associated directive **142**. It should be noted that the next operation information will differentiate between protocol data units for the first **152** and second virtuals switches **154**, and may include header modification or truncation of the protocol data unit **140** prior to forwarding. The memory buffer **106** works in conjunction with the forwarding processor **108** and the outbound interface **110** to accomplish this task as preferably as a real-time operation.

The media header information typically includes at a minimum the encapsulation type, protocol type, frame type, media destination, and source route information. This information is used by the modifier device **126** to add next operation information **142** which specifies a particular operation such as route, bridge, or source route bridge to perform on the received protocol data unit. It should be noted that a media destination may be multicast, a unicast match, or a unicast non-match destination.

In order to accelerate the forwarding of a received protocol data unit **140**, the identifier **122** preferably is configured to determine the media header information after having received only a portion (i.e., the first several bits or bytes) of the protocol data unit **140**. Similarly, the modifier device **126** preferably is configured to add next operation information **142** to the media header information after having received only a portion of the protocol data unit **140**. Both of these optimizations are particularly important when manipulating large protocol data traits which are simultaneously received from various incoming interfaces **102**.

The preprocessor **104** preferably includes a mechanism for aligning the contents of buffer memory **106**. This can be accomplished by the modifier device **126** padding bytes of data to the protocol data unit **140** such that the header

information is aligned on optimal boundaries. The reasons for doing this optimal boundary alignment are discussed in the following sections.

The preprocessor **104** identifier **122** preferably includes an address lookup mechanism for obtaining various addresses required by the preprocessor **104** through the use of a content addressable memory **128** (CAM) located in the forwarding processor **108**. The addresses can be obtained through several types of algorithms. For example, a network destination address of the protocol data unit **140** can be compared to a predetermined list of known network destination addresses. Also, a media destination address of the protocol data unit **140** can be compared to a predetermined list of known media destination addresses. Further, a media source address of the protocol data unit **140** can be compared to a predetermined list of known media destination addresses. Furthermore, a media source address of the protocol data unit **140** can be compared to a predetermined list of known media source addresses.

The preprocessor **104** identifier **122** also preferably includes a source route bridge destination lookup mechanism for checking for specifically routed protocol data unit **140**, finding next local area network identifier in a source route of the protocol data unit **140**, and comparing the next local area network identifier to a predetermined list of known local area network identifiers by utilizing the CAM **128**.

The present invention also can be described in reference to a device-implemented method steps **200–218** shown in FIG. **6** to deliver protocol data units within a first **186** and a second **188** virtual closed user group on a shared communication medium **158** in a communication system **190** (i.e., shown in FIG. 4). This delivery method includes maintaining **202** a database of all destination identifiers currently reachable for delivery of protocol data units within the communication system. Access to this database is limited **204** such that each virtual closed user group only has access to specific destination identifiers owned by that particular virtual closed user group. Incoming traffic is assigned **206** to each virtual closed user group based on an access policy that is separately specified in each virtual closed user group. The access policy may be as simple as having all members with last names beginning with the letters A-I being placed in one group and all others being placed in another. Further, the access policy may be more complex. For example, member group assignment could be based on an IP address, plus a known security clearance level, and a time of day to which access is limited (i.e., only during standard working hours).

At this point, the operations are split between the first and the second virtual closed user groups such that any further processing is performed separately. Data bits within a protocol data unit received from a data interface on the shared communication medium are examined and modified **208** wherein the protocol data unit has a unique destination identifier associated with the first virtual closed user group to form an associated directive for that protocol data unit. In a similar manner, data bits within a protocol data unit received from a data interface on the shared communication medium are separately examined and modified **208'** when the protocol data unit has a unique destination identifier associated with the second virtual closed user group to form an associated directive for that protocol data unit. Each member of the second virtual closed user group also has a unique destination identifier.

Also, verification **210** and **210'** that each destination identifier in a protocol data unit on the shared communication medium is currently reachable by the particular virtual closed user group for delivery is required This verification may be accomplished through a lookup in the database prior to completing delivery of the protocol data unit to the associated destination identifier. If a destination identifier is not found in the database, three possible actions can be taken, including: dropping the associated protocol data unit so that no delivery occurs, forwarding the associated protocol data unit to another destination identifier, or adding this non-verified destination identifier to the database. This last possible action also has other benefits in that any particular device capable of communicating on the shared communication medium can be a member of either virtual closed user group by simply adding **212** or **212'** a destination identifier associated with the particular device newly attached to the communication network to the database and associating the device with the first or the second virtual closed user group, respectively.

Subsequently, the first virtual closed user group modified protocol data unit is delivered **214** to another member of the first virtual closed user group after verifying that the first virtual closed user group member destination identifier is currently reachable. In addition, the second virtual closed user group modified protocol data unit is delivered **214'** to another member of the second virtual closed user group after verifying that the second virtual closed user group member destination identifier is currently reachable.

These device-implemented steps **200–218** preferably are performed such that all predetermined physical layer, link layer, and network layer access protocols used to communicate protocol data units over the shared communication medium are preserved. In other words, no changes to protocol like IP, ATM, OC-12 or the like are necessary to implement the present invention, because these steps are seamlessly integrated with this access protocols.

In addition, the device-implemented steps may include a step of providing **216** a virtual link between the first and the second virtual closed user group. This virtual link can be used to deliver protocol data units from the first virtual closed user group and to the second virtual closed user group. The virtual link includes a first end and a second end of a data path on the shared communication medium. Also, each virtual link end includes a data port in a different virtual closed user group. A shared memory can be used as the shared communication medium to provide the virtual link. Alternatively, a high data transfer rate link which spans a geographic distance between the first and the second virtual closed user group can be utilized to provide the virtual link. A filtering process can be performed on the virtual link such that protocol data units communicated in the virtual link data path are filtered according to an access policy that is separately specified in each virtual closed user group. It will be appreciated by those skilled in the art that another virtual link may be provided to deliver protocol data units from the second virtual closed user group to the first virtual closed user group.

Mode for Carrying Out the Invention

The following sections describe in detail the features required for the virtual router/switch. It should be noted that the packets described herein refer to protocol data units which can take the form of a packet, cell, or frame of data. The Virtual Router feature provides a means of segregating Upper Layer Protocols and their ports into several groups or "virtual" routers. These virtual routers can function almost totally independent of one another preventing traffic mix, or allowing controlled access amongst one another.

The following describes the customer viewable aspects of the Virtual Router. This includes the Virtual Router (VR) component and all of its immediate subcomponents. In particular, the following details regarding the VR are described:

Component Definitions;

Provisionable attributes;

Operational attributes;

Provisioning commands; and

Provisioning procedures.

Some of the primary benefits of the Virtual Router include:

Multiple Virtual Routers per platform

Service Providers: Provides ability to share the resources of one platform across many customers. Obvious benefits include cost reduction and integrated maintenance.

Large Enterprises: Provides ability to segregate local traffic and resources between organizations while sharing the more expensive long haul resources among organizations. Again benefits here include centralized maintenance and reduced cost.

Controlled interaction between Virtual Routers—The VR provides for traffic segregation and controlled integration. Integrated traffic can be precisely controlled regarding its source, destination among many other possibilities.

Single maintenance platform—Provides a constant and integrated management interface for each Virtual Router on the platform. This simplifies the management of the set of VRs.

The Virtual Router (VR) provides the customer with the ability to segregate router traffic and maintenance into what appears to be separate physical routers. This is accomplished by the creation of multiple instances of the VR component directly under CASROOT.

Each instance of the VR component provides support for multiple network protocols, routing protocols and media devices to operate virtually independently of each other.

The VR system includes the VR component and many other subcomponents. These sections provide the complete specifications of the following components:

Virtual Router;

Protocol Port; and

Memory Management.

These sections describe the subcomponents under the VR, including the number and type of subcomponents which can exist directly under the VR and Protocol Port (PP) Components.

The Virtual Router component resides directly under CASROOT. Multiple instances of the VR component may exist. Components which reside under the one VR component are independent of components under another VR. The network protocol and protocol port components reside under the VR component. This allows multiple instances of protocol stacks to exist on one platform. The network protocol component represents an instance of that particular protocol and all of its configured data.

Protocol Ports (PPs) are created under the VR and represent a logical interfaces to a network. There can be multiple protocol ports under each VR, each having any number of protocols enabled. To allow for the processing of particular OSI Network Layer Protocol packets on a PP, a Network Protocol Port subcomponent is created under the PP. A CAS linkage, from the PP to a Media Application component, is used to associated the logical PP to a physical media. Each PP must be linked to a Media Application component. The linkage between PPs and Media Application components is a one to one linkage.

An example of a component hierarchy including three Virtual Routers and multiple protocol stacks and protocol ports is shown in FIG. **18**. The VR has many subcomponent, many of which also have subcomponents. FIG. **19** is a diagram which shows the entire component hierarchy under the VR. The bold or darker tone components are fully defined in these sections.

VR PPs support both LAN and WAN media. For LAN media, the PP is linked, via a CAS linkage, directly with the LAN Application component. This linkage defines the physical port which the PP is associated with.

For WAN media, the linkage from PP to WAN Application varies depending on the Link Level Protocol (LLP) type. This feature supports two very different types of LLP types. One type of LLP is a Point to Point LLP. A Point to Point (PPP) LLP type provides a 1 to 1 connectivity model. That is each end of the link supports only one end destination. The WAN Applications which use this model include: Point to Point Protocol (PPP) and Vitalink Control Protocol (VCP). For these LLP types, the PP is linked directly with the Application component, e.g. PPP or VCP.

The second type of LLP supported by the VR system is the Multi-point LLP. The multi-point LLPs allow for one interface to be multiplexed into multiple data carrying pipes, each capable of supporting multiple end destinations. The WAN Applications which use this model include: Frame Relay Data Terminal Equipment (DTE), X.25 (DTE), and Switched Multimegabit Data Service (SMDS) (DTE). For all of these LLP types, the PP is linked with a RemoteGroup subcomponent which resides directly under the Application Component.

Networking Protocols (e.g., IP, IPX, etc.) are supported via CAS components which reside directly under the VR component and a Network Protocol Port components which resides directly under the PP component. There can be at most, one instance of each Network Protocol component under each VR component and one instance of each Network Protocol Port component under each PP. These subcomponents provide the functionality required to provide packet processing support for each Network Protocol Type. It is required that for each Network Protocol Port which is provisioned under a PP, that the associated Network Protocol component exist under the VR.

IP is supported per VR with an Ip component and an IpPort component. There can be at most one Ip component under each VR and one IpPort component under each PP. Support for multiple IP addresses per IP protocol port is provided by a subcomponent under the IpPort component. This component contains a list of "secondary" IP addresses which are associated with the IpPort. Routing Protocols (e.g., OSPF and Exterior Gateway Protocol (EGP)) are supported via CAS components which reside directly under the IP subcomponent. Thus the creation of a IP component also provides for the creation of these routing protocols.

IPX is supported per VR with an Ipx component and an IpxPort component. There can be at most one Ipx component under each VR, and one IpxPort component under each PP.

Appletalk is supported per VR with an Appletalk component and an AppletalkPort component. There can be at most one Appletalk component under each VR, and one Appletalk Port component under each PP.

Decnet is supported per VR with an Decnet component and an DecnetPort component. There can be at most one

Decnet component under each VR, and one DecnetPort component under each PP.

Bridging is supported via the Bridging CAS component which resides directly under the VR component. There can be at most, one instance of a Bridging component under each VR component. The creation of a Bridging component under a VR component, allows for the creation of a BridgePort subcomponent under the PP component.

Cluster bridging provides the ability for a group of LAN ports to be bridged for all media, while treating the group of ports as one port which can be used for Telnet and management purposes. Cluster bridging is supported by the VR using two existing components and one new component. The existing Bridge component is still used under each VR to perform the bridging activities. The BridgePort component is used to enable bridging on a PP. This component is used to enable bridging on a standard bridged port and to enable cluster bridging on a cluster bridged protocol port. A standard bridged port is linked to a LAN application which identifies the physical interface which is being bridged. A cluster bridge PP is linked to a new component called a ClusterBridge. This component represents the cluster bridge logical interface and it is what distinguished a PP as being a cluster bridge PP. The ClusterBridge Application component resides directly under the VR. A PP linked to a ClusterBridge Application component (versus a LAN application component) represents a ClusterBridge port. This port acts as a gateway for all bridged PP traffic to routing, and from routing to all bridged PPs.

Multiple cluster bridge ports may be defined, using multiple instances of the ClusterBridge application. Each bridged port can belong to at most one ClusterBridge. Each BridgePort, including cluster bridge PPs, have a domain attribute which identifies the bridge domain that is used by bridging. The bridge domains must be unique per cluster bridge Port. The bridge domain is what determines which bridge ports are associated with specific instances of the ClusterBridge. For example, all bridged ports with domain 5 are associated with the single cluster bridge port with domain 5.

FIG. 20 is an example of multiple cluster bridges, each associated with one bridge PP. Notice the domain numbers for the bridged ports and the cluster bridge ports. Bridged PP/0 is associated with cluster bridge PP/1 and bridged PP/3 is associated with cluster bridge PP/2.

SNMP is one method which can be used to manage the physical and virtual router platform. This method requires VR support for the SNMP agent(s) which reside under Access Control. The VR provides mechanisms which allow each VR to be managed virtually independently of one another. The SNMP support is provided on a per VR basis with respect the VR components and software. If SNMP support is required, an SNMP component can be provisioned under the VR component. This component contains the ifTable and SnmpSystemGroup. The ifTable includes an ifTable entries for each interface in the VR The SnmpSystemGroup contains generic configuration information per VR for SNMP.

The SNMP administrative status and OSI state must both be supported since it is desired to support both SNMP and OSI. The OSI state is purely an operational attribute while the SNMP admin state is a provisionable attribute. This is the only difference between the SNMP admin state and OSI admin state. The issue here is how these states can affect each other and how they are to be dealt with. The particular state dependencies are well known in the art and for simplicity will not be repeated herein.

The Packet Control Facility (PCF) provides a mechanism to monitor and control the flow of packets through the router platform. This facility is provided on a per VR basis. A PCF component can be produced under the VR which provides the mechanisms and functionality to create, and apply PCF filters.

The following section will describe the memory management for the VR. The major concern for memory management (MM) is to prevent a VR or one of its Network Protocols from using so much memory that it begins to affect the other VRs or protocols adversely. A required Memory-Management component is created under the VR which allows for the configuration of Memory Management. The memory management component allows for the management of both the VR and its Network Protocols.

The memory management for the VR is primarily designed to prevent one VR from obtaining an amount of memory which adversely affects other VRs. MM attributes allow for a percentage of total heap space to be provisioned as the maximum amount of memory which the VR and its subcomponents can obtain. The total heap space for the system is calculated during initialization. It includes the amount of heap space remaining after the standard port of system initialization and prior to the initialization of the provision dependent processes. There are no restrictions on the percentage of heap space any VR can allocate. In fact the total heap space for the platform may be overallocated. For example, three VRs VR/0, VR/1 and VR/3 may each allocate 50% of the heap space on the platform. Allowing this has some apparent advantages and disadvantages. An apparent disadvantage is that a VR may be denied memory before it reaches its maximum memory amount. However, the maximum memory amount is just that, a maximum and it is not to be implied that it is also a guaranteed minimum. An advantage of over allocating memory is that it allows for VRs to share heap space. on a first come first serve basis.

When a VR attempts to allocate an amount of memory which would exceed the maximum allowed, it will be denied. In addition to the denial of memory, an alarm will be triggered. The alarms are filtered such that no more than one alarm per VR will exist per period of time. This period of time is a provisional attribute.

The MM for a VR can be a very powerful tool in controlling the affects of an errant VR on other VRs. In order to efficiently provision the memory maximums, the user must know how much memory a VR is using under normal situations. To provide access to this data, operational attributes are provided which lists the amount of heap space is currently in use per VR.

Because the maximum share of memory is a provisioned attribute, it can be adjusted to include more memory or less memory. If the provisioned amount is increased, MM automatically adjusts the amount allowed to the VR and Protocols. If the amount of memory is decreased, no action is taken by MM to automatically retrieve memory from VRs which now exceed that amount. If it is necessary to reduce the amount of memory used by a VR or protocol, the VR can be administratively disabled and enabled. This restarts the VR, thus requiring them to adhere to the new provisioned values for maximum heap space usage.

The memory management for Network Protocols work much like the MM for VRs. Separate attributes within the same MM allow for Network Protocol memory management. A provisioned attribute for each protocol determines the percentage of heap each protocol can use. However this share, unlike the VR share, is in terms of VR heap space versus total heap space. So if a VR has a provisioned fair

share of 50%, and a protocol has a fair share of 50%, the protocol has at most, the capability of getting 25% of the total heap space of the system.

The Network Protocol memory management includes the same type of alarm reporting and current memory usage counts as described above for the VR. Similar to the VR situation, if the provisioned share of memory used by a protocol is decreased, a protocol could be in violation of it share of the memory. Although it has more memory than it should have, all subsequent requests for memory will be denied. If it determined that this is a problem, the situation can be corrected by disabling the protocol and re-enabling it. Otherwise the protocol may be in violation until either the protocol reduces it memory consumption for another reason, or the system reboots.

A MemoryManagement component exists under the VR to contain all the attributes associated with VR and Network Protocol memory management. The MemoryManagement component is a required subcomponent, which has default values for each of its attributes.

These sections detail the ability to support multiple VRs on one router platform. In addition, it also provides a means of internally connecting these VRs such that they can transmit packets between one another. This capability is be referred to as inter-VR support and is shown in FIG. 21.

In the standard intra-VR case, a VR protocol port is linked to a media application component. This linkage explicitly defines the physical port, and implicitly the FP, the logical port is associated with.

In the inter-VR case, the PP has no physical port or media which connects the two VRs. A mechanism must be devised to replace the functionality provided by the physical port and connection. The two most obvious alternatives were to either connect the VRs via a Virtual LAN media or a Virtual Point to Point Link media.

An inter-VR connection can be achieved by creating two protocol ports, one on each VR, and connecting them via a Virtual Link component. The two protocol ports connected to the Virtual Link component are considered inter-VR (IVR) PPs. Although the Virtual Link is a logical mediun versus a physical medium, it is what is used to interconnect the two VRs.

Provisioning a Virtual Router consists of provisioning multiple components, most notably the Virtual Router (VR) as well as its subcomponents (Upper level Protocols, SNMP, Bridges and Protocol Ports). The Virtual Router is used to logically separate sets of protocol ports and protocols into what appear to be and act like distinct routers/bridges. Most of the components required to operate the platform as a router or bridge either reside under the VR or are linked to the VR via a CAS linkage. This allows the creation of multiple instances of these components on a per Virtual Router basis.

The Virtual Router (VR) component resides directly under CAS Root. The base software set allows for the creation of only one VR. However since part of the optional software set, allows for multiple VRs, the VR must always have an instance identifier. Preferably the maximum number of VRs allowed is 16; however, this number is totally arbitrary and changing it to a larger or smaller number should be a trivial operation. The identifier for the VR is a string of up to 20 characters.

To allow for inter-VR packet processing, VirtualLinks are used. A VirtualLink allows for the connection of at most two VRs. The VirtualLink component resides directly under CAS Root and can be linked to two Protocol Ports through a provisioning procedure. CAS checks ensure that both linkages are established and that they are between two different VRs.

The following caveats are a result of the implementation of the VR on this preferred embodiment router platform:

The provisioning system is not VR centric—Because only one edit view exists for all CAS components, multiple VR provisioning sessions are not possible. Additionally, CAS command security between VRs is not supported.

Resource Contention—Hardware and Software resources are shared across VRs, thus certain conditions on one VR may affect the performance of another VR.

Each instance of a Virtual Router requires the creation of many components and processes. This obviously has a significant impact on memory usage. Depending on may variables, including the number of protocols supported per VR, the number of Protocol Ports created and the size of the dynamic routing tables created based on the network design, the memory usage may vary. Thus the number of VRs which can be successfully provisioned and made operational is nondeterministic. Several features which, if provisioned, may significantly affect the amount of memory used. The primary memory user is Multiple Virtual Routers. Each instance of a VR uses a significant amount of memory depending on the number of protocols, protocol ports and media types being used.

The use of the following features may have create some level of performance degradation compared to the optimal performance characteristics:

ClusterBridging—Because cluster bridging requires both a bridging and a routing step, the performance regarding forwarding a packet via a ClusterBridge port can be reduce to half the normal rate.

Inter-Virtual Router Packet Forwarding—The process of forwarding a packet from one VR to another includes forwarding a packet out a IVR PP for one VR, in the IVR PP for the destination PP and finally out a physical PP. This process includes at least two forwarding table lookups, and one mapping from the inbound VR to the outbound VR. This process in some cases can be split across FPs, thus incurring additional performance penalties.

The following sections are the System Description for the Virtual Routing System which provides for the creation and execution of multiple Virtual Routers on a physical router platform. It also includes what is called "Protocol Glue". This is the software which provides a common set of interfaces and support mechanisms for the Network Protocols to run within.

The Virtual Router (VR) feature has two major parts. The first part, is to provide the functionality required to support multiple Virtual Routers on a single physical router platform. The second part is to provide a common set of interfaces and support mechanisms for the network protocol processes.

The VR is primarily intended for customer premises applications where a single chassis may operate separate networks for multiple clients. This allows an internetworking service provider to support multiple clients, multiple networks while sharing hardware. Traffic can be cross-connected between VRs using Virtual Links. Virtual Links act like Point to Point media for InterVirtual Router connections.

In the following sections, Network Protocols, ULPs, and Protocols are used to refer to the same thing, (i.e., IP, IPX, Decnet, AppleTalk, etc.). The terms Media Applications and Applications are used to describe the Media component

processes (i.e., Ethernet, FDDI, TokenRing, FrameRelay, etc.). The term "event" is used extensively throughout this section. An event identifies any external stimuli to a process or object. An event can be a Process Environment (PEV) message or a function call. The term Protocol Port (PP) is used extensively and it refers to PPs in general. There are many different types of PPs, used in different components and processes in addition to the PP component itself. If a specific PP type is being referred to, its specific type is used.

The Virtual Router system consists of several classes. The three major classes which represent PEV processes are the VirtualRouterProcess, ForwardingAgentProcess and the NetworkProtocolBaseProcess classes. These processes are shown in FIG. 22 along with the sub-processes which they interact with. In particular, the following sub-processes are shown:

Component Administration System (CAS);

Packet Control Facility (PCF);

Process Control System (PCS);

Component Name Server (CNS);

Global Stats Manager (GSM);

Global Cache Manager (GCM);

Local Cache Manager (LCM); and

Local Stats Manager (LSM).

The VirtualRouterProcess class encapsulates the control processor (CP) data and functionality associated with each instance of the VR. The VR processes reside on the CP. One VR process is created by CAS for each VR provisioned on the platform. The VR process creates network protocol processes for each provisioned Protocol under the VR.

This class provides the mechanisms required to handle all CP related activities as required by the protocol processes, the protocol ports and other objects which require VR support.

The ForwardingAgentProcess class provides the functionality required by the VR system on each active Logical Processor (LP) on the router platform, including the CP. This class provides the process which acts as the VRs agent on the Logical Processors (LPs). The VR uses these processes in order to provide each LP with forwarding data and other initialization required by the protocol.

The NetworkProtocolBaseProcess Class provides a common platform for which each network protocol process is built on. This platform provides a common set of interfaces to all processes interacting with the protocol processes. Having a common interface for all network protocols reduces the effort and complexity of system components which interface to all the protocol types.

Protocol registration is initiated by the Network Protocol processes. It is an indication to the VR that the Protocol is active and wishes to begin receiving locally addressed packets. In addition to this, it provides the VR with Protocol specific forwarding information (transparent to the VR), which is required on the LPs for packet forwarding. This data is made available to the protocol specific data path objects on each LP via the VR/Forwarding Agent (FA) processes. Included in this Protocol registration message is the forwarding information, the process ID of the process which is to receive local packets for the protocol, and an indication of the state of the protocol (active/inactive). The protocolState is required since this message is being used for registration and deregistration.

Protocol processes require notification that the media interface is available prior to initiating a bind request. This notification originates from the Media Application component once instantiated.

Media Application processes (e.g. ENET, FrameRelay, etc.) are created on LPs (in the general case) and on the CP (for the ClusterBridge and Inter-VR Link cases) during CAS provisioning. These applications have CAS linkage to a PP. When a Media Application process becomes active, it registers with the Virtual Router on the CP. Included in the registration message is the data required by the forwarding Data Manipulation Engine (DME) (i.e., frame handler) regarding the physical interface type, maximum packet size, etc.

Upon reception of the message, the VR saves the data regarding a PP in the VRs PP object. This data is sent down to the FA and made available to the forwarding software. In addition to this, the VR sends a mediaAvailability message to each protocol which is provisioned trader this PP. Once this message is received by the protocol processes, they can bind to the PP.

Protocol Port Binding is a procedure used by a network protocols to enable or start packet processing on a PP for a particular protocol type. For example, prior to IP binding to PP/5, all IP packets on PP/5 are either bridged (if bridging is active) or dropped. Once IP binds to PP/5, IP packets are processed by the IP forwarder.

Because the forwarding software is on the LP and the network protocol processes are on the CP, the VR provides a mechanism to support protocol binding. Binding is initiated by the network protocol process via a bind request delivered to the VR via a PEV message. The VR then determines which LP(s) the PP resides on, and forwards the Bind request to the appropriate FA(s). Bind data, passed by the network protocols, is used during the creation of the Bind Table Manage (BTM) on the LP. A BTM is required on each LP which can process inbound packets. For PP which have physical interfaces, the BTM is required on the LP where the physical interface resides. For PPs that have no physical interface (e.g. ClusterBridge) and PPs that have multiple physical interfaces more than one LP requires a BTM.

The FA, upon reception of the bind request, creates a BTM (if necessary) and delivers the bind request to it. In addition, a ProtocolPort object is created using the data included in the bind request by the protocol process. This ProtocolPort provides all the information required by the forwarding software to process packets on this interface.

Before network protocol DMEs can be used on the LP, they must be created. A DME must be created on each LP for each bound Protocol. The VR and FA assist in the creation of these DMEs.

Protocols have semi-global data which is required to forward packets. This is the data which is not PP specific, but protocol specific. This data is passed to the VR in the protocol registration message. This data is forwarded to the LPs and a pointer to this data is passed to the function which is called to do the protocol LP initialization.

Each protocol must provide a function (a forwardingEventHandler) which can handle forwarding events initiated by the protocol process. This function must be able to handle each of the protocolForwardingEvent types (ProtocolEnabled, ProtocolDisabled and Protocol Updated). The ProtocolEnabled event triggers the initialization of the protocols LP forwarding data (including the creation of the protocols DME). The function is called prior to creating the first PP for a particular protocol on each LP. This function, which is to be provided by the network protocol, must be made available on the LP. As the saying goes, "cverything that goes up must come down". When a protocol deregisters, a function is called to disable the initialization done for the protocol and remove the DME for the protocol.

To enhance the performance of forwarding packets on the router platform, all packet processing required to forward a packet on some LAN media is done on the inbound board. Because of this, the forwarding software on an LP not only requires information about all the PPs on the LP, but also for all the PPs for which it can transmit out on other LPs. This PP information, distributed across multiple LPs for a port on one LP is called "Distributed Protocol Port Data". This data is required on each LP prior to forwarding the first packet. The data is forwarded to the LPs when the LP became active and the Media Application registers with the VR. The data is sent to the LP for each PP on the VR.

When the distributed PP data arrives on the LP, the FA creates a ProtocolPort object. This object contains all the data necessary for the forwarding software to forward a packet.

The Protocol Port (PP) is a very significant part of this system. It represents a logical interface to a network. The term PP has many different connotations. There is the PP component which resides under the VR. There is also a Network Protocol Port component which is specific to a particular network Protocol type. In addition to these PP components there are VR PP objects, Network Protocol Port objects and Forwarding Agent Protocol Port Objects. Each of the PP objects require a slightly different view of the PP. For example, the VR is interested in the protocol independent portion of the PP. The protocol processes are interested in the protocol dependent portion of the PP, and the forwarding DME software is interested in yet a different portion of the software (in fact, forwarding software on LPs that contain the physical interface associated with a PP require different information than those on the other LPs). In addition to all of this, the PP information is distributed across multiple processes, not to mention multiple processors.

The following protocol port object types exist, including: VrProtocolPort, NetworkProtocolPortBase (also one derived version for each Protocol type), FaInboundProtocolPort, and FaOutboundProtocolPort.

The VrProtocolPort provides the data and functionality required for the PP by the VR process. This includes all PP data which is common across all protocols. This data can be broken down into two categories, inbound and outbound. The inboundPpForwardingData is a structure of common data required by the inbound forwarding software. The outboundPpForwardingData structure contains data common to multiple protocols required by the outbound forwarding software. These two structures along with methods which are used to update the contents of them, make up the VrProtocolPort.

The NetworkProtocolPortBase class provides protocol processes with a base class which is be used to encapsulate protocol specific forwarding data. This forwarding data, in terms of protocol specified objects, is turned into contiguous bytes of data, transmitted to the LPs and reincarnated into their original form again. Once back into the object form, they can be used by the forwarding software to forward datagrams. This base class is meant to provide a common interface to the VR so that the data can be turned into contiguous bytes which are sent to the FAs, where they are reincarnated back into objects of their original form.

The FaOutboundProtocolPort class provides the data and functionality required on LPs which need outbound PP data in order to forward the packet. This class contains the protocol independent outboundPpForwardingData structure from the VR as well as the protocol specific outboundForwardingData provided by the protocol process.

The FaInboundProtocolPort class is inherited from the FaOutboundProtocolPort class. This class provides both the inbound and outbound data required forwarding software. This class is used on LPs where inbound packets can be processed, thus requiring inbound forwarding data in addition to the outbound forwarding data. In addition to the outbound data provided by the base class (FaOutboundProtocolPort), this class contains the protocol independent inboundPpForwardingData structure from the VR as well as the protocol specific inbound forwarding data provided by the protocol process.

The following section describes the initialization procedures provided by the VR system on both the CP and LP.

The Virtual Router requires no pre-provisioning CP initialization. All VR initialization is performed during or after VR provisioning. The Virtual Router system requires system initialization on the LP. The Virtual Router system requires a process on the LP which acts as an agent for the VR on the LP. This agent process, called the Forwarding Agent (FA), is created during LP initialization. This process immediately registers itself with the Component Name Server (CNS) so that other processes have access to its PID.

The following section describes high level details regarding the processing of messages from CAS. The VR process handles CAS messages for itself and all of its subcomponents. When a provisioning message is received by the VR for a subcomponent which requires a process, the VR creates the process, and forward the CAS provisioning message to that process. In this case, it is the responsibility of the subcomponent process to handle the provisioning message appropriately and provide the acknowledgment to CAS. CAS messages which are addressed to the VR or to components which do not have processes are processed by the VR.

A Virtual Router Process is created by CAS for each provisioned VR. Once the process is created, CAS sends all provisioned information for a VR to the VR process. The VR creates a process for each provisioned protocol. Once the process is created, the VR sends all the provisioning messages for the protocol to the protocol processes.

The VR system primarily consists of two processes, the VR process and the FA process. The VR process is created by CAS as the appropriate CAS messages are delivered. The FA process is created by fixed process initialization on each LP. Forwarding packets for a particular protocol over any media requires a certain amount of initialization. The initialization procedure required can be different depending on the type of media a PP is connected to. This section discusses the initialization required by each of the various media types and the packet forwarding process associated with them. For the standard case (i.e., LAN media), the per port initialization required prior to packet forwarding includes Virtual Router Creation and Provisioning as shown in FIG. 23. The following steps must be accomplished.

A1. CAS Create and Provisioning messages—The VR process is created by CAS and all VR provisioning messages (including all of those for VR subcomponents) are sent to this process.

A2. Each CAS provisioning message received by the VR for protocol subcomponents (including PCF and SNMP) are forwarded to those subcomponents processes. (The standard CAS provisioning message is used and forwarded to the protocol processes)

A3. The ProvDone message from CAS indicates that CAS has finished sending CAS provisioning message to the VR for this CAS session.

A4. The VR sends a ProvDone message to each of its subcomponents processes indicating to them that CAS has finished send provisioning messages for this CAS session. (The standard CAS provisioning message is used and forwarded to the protocol processes)

A5. Prior to giving up control of the Execution Engine (EE), the VR sends a PEV message to the CNS registering its process ID. (The standard CNS registration message type is used)

A6. After the Protocol Processes receive the ProvDone message, it can register with the VR. This registration indicates the protocols acceptance of local packets and the process ID which is to receive them.

In addition, LAN Media Application Creation and Initialization must be accomplished prior to packet forwarding as shown in FIG. 24. The following steps must be accomplished.

B1. CAS Create and Provisioning messages—The Media process is created by CAS and all provisioning messages are sent to this process.

B2. The ProvDone message from CAS indicates that CAS has finished sending CAS provisioning message to the Media for this CAS session.

B3. The Media application requests the process ID associated with the VR process. A linkage attribute in the Media Application defines the VR which is associated with this media. (The standard CNS registration message type is used)

B4. Some time later, the process ID associated with the request esVR is returned to the Media Application process. This event does not occur until event A5 has occurred. (The standard CNS registration message type is used)

B5. The Media Application registers with the VR. This notifies the VR of the Media's availability and process ID.

Also, Forwarding Data Distribution—LAN Media must be accomplished as shown in FIG. 25. The following steps must be accomplished.

C1. This event depends on event B5 having occurred. The Create Protocol Port event sends the generic protocol port information associated with one or more protocol ports to the relevant set of LPs. If this is the first PP which is created on this LP, PP information for all registered PPs are also sent to this LP. If this is not the first PP which is created on this LP, only the PP information for the registered PP is sent to this LP. The relevant set of LPs is all LPs which have a physical interface in the VR.

C2. Call the Media ForwardingEventHandler—Pass the Media Forwarding Data. The Media FEH creates the Media DME and any other Media required entities.

C3. If the LP which receives the create PP message is in the inboundLpSet, a FaInboundProtocolPort object is created, otherwise a FaOutboundProtocolPort object created.

C4. If the LP which receives the create PP message is in the inboundLpSet, create a BindTableManager and a fast BindTableManager and store pointers to then in the FaInboundProtocolPort object.

C5. If the ifEntryRegistration field is set in the faCreateProtocolPort message, space is allocated in Local Stats Manager (LSM) for the ifEntry. (LSM provides the message for this event.)

C6. Update the physicalPortInfo Structure—Set the pointer to the FaPP which is associated with the physical channel.

Also, Protocol Binding—LAN Media must be accomplished as shown in FIG. 26. The following steps must be accomplished.

D1. This event can only occur after event B5 occurs. B5 is the MediaRegistration event which indicates the media for the specified PP is available. This event notifies each protocol which is provisioned under the specified protocol port of the media's availability.

D2. The protocol, upon receipt of this mediaAvailability message can bind to a PP. This binding determines the actions which should occur for packets received for the specified protocol type.

D3. The BindProtocolPort event is used to send the bind information associated with the individual protocols to the appropriate LPs. Included in this bind information is Protocol Forwarding Data and Protocol Specific PP Forwarding Data. Protocol Forwarding Data includes the data specific to each protocol which is required by the forwarding software to forward packets. this information was included in the Protocol Registration (event A6). The Protocol Specific PP Forwarding Data includes forwarding data required by each protocols forwarding software which is associated with an individual PP.

D4. If the Protocol has not yet been enabled on this LP, the appropriate Protocol ForwardingEventHandler is called.

D5. The following events can occur in any order. Update the PP forwarding data in the FaPP. Also, Register with the Local Cache Manager (LCM) (Message is to be provided by the LCM).

D6. Update the state of the BTM—Update the forwarding state for the BTM.

At this point, packets received by the Frame Source can be forwarded. The forwarding process works as shown in FIG. 27 (i.e., Packet Forwarding—LAN Media).

E1. The dispatcher pulls a packet off the inbound packet queue and calls the appropriate Media frame source with the physical channel number which the packet came from.

E2. The Frame Source gets a pointer to the Inbound PP object via the physicalPortInfo structure maintained by the FA.

E3. The inbound Media DME is called to process the packet.

E4. The protocol Forwarding DME is determined via the BTM. The inbound PP object has a pointer to the BTM.

E5. The protocol forwarding DME is called. The appropriate protocol processing is done.

E6. The protocol forwarding DME does a cache lookup to determine the outbound PP for which the packet is to be transmitted out. (Part of the cache entry is a pointer to the FaProtocolPort object)

E7. The outbound Media DME is retrieved from the FaProtocolPort object.

E8. The outbound Media DME is called to process the packet.

E9. The appropriate media processing is done and the packet is transmitted out the interface.

The major difference between the standard LAN case and the standard WAN case (not including PPP/VCP in the standard WAN case) is that the WAN case has its own mapping table from inbound packet to inbound PP. In the LAN case, the mapping to a PP is based on the inbound physical channel. For the RemoteGroup WAN case, the mapping to the PP object is based on data inside the packet (e.g. Frame Relay DLCI or Data Link Connection Indicator). The WAN component must initialize a table based on the mapping data to a ppId. The WAN media has a mapping from DLCI to PP instance ID, but they do not know what the PPID is associated with the PP instance. This is required to get a pointer to the PP object via the logicalPortInfo table supported by the FA. The WAN media gets the PP instance ID to PPID mapping from the VR during the createProtocolPort event. If the createProtocolNotification flag is set, the FA sends a notification to the WAN which includes the PP instance ID and the PPID.

Shown in FIG. 28 is the PP initialization for the multi-point WAN media (i.e., Media Application Creation and Initialization—Multi-point WAN). The following steps must be accomplished.

B1. CAS Create and Provisioning messages—The Media process is created by CAS and all provisioning messages are sent to this process.

B2. The ProvDone message from CAS indicates that CAS has finished sending CAS provisioning message to the Media for this CAS session.

B3. The Media application requests the process ID associated with the VR process. A linkage attribute in the Media Application defines the VR which is associated with this media. (The standard CNS registration message type is used)

B4. Some time later, the process ID associated with the request esVR is returned to the Media Application process. This event does not occur until event A5 has occurred. (The standard CNS registration message type is used)

B5. The Media Application registers with the VR. This notifies the VR of the Media's availability and process ID. The createProtocolPortNotification field is set to ensure the Media Application is notified of the PPID when it is created on the LP.

Also, Forwarding Data Distribution—Multi-point WAN must be accomplished as shown in FIG. 29. The following steps must be accomplished.

C1. This event depends on event B5 having occured. The Create Protocol Port event sends the generic protocol port information associated with one or more protocol ports to the relevant set of LPs. If this is the first PP which is created on this LP, PP information for all registered PPs are also sent to this LP. If this is not the first PP which is created on this LP, only the PP information for the registered PP is sent to this LP. The relevant set of LPs is all LPs which have a physical interface in the VR.

C2. Call the Media ForwardingEventHandler—Pass the Media Forwarding Data. The Media FEH creates the Media DME and any other Media required entities.

C3. If the LP which receives the create PP message is in the inboundLpSet, a FaInboundProtocolPort object is created, otherwise a FaOutboundProtocolPort object created.

C4. If the LP which receives the create PP message is in the inboundLpSet, create a BindTableManager and a fast BindTableManager and store pointers to then in the FaInboundProtocolPort object.

C5. If the ifEntryRegistration field is set in the faCreateProtocolPort message, space is allocated in LSM for the ifEntry. (LSM provides the message for this event.)

C6. Update the logicalPortInfo Structure—Set the pointer to the FaPP which is associated with the PP identifier (PPID).

C7. Send a CreateProtocolPort notification to the Media Application. This notifies the Media of the PPs creation and the PPID associated with it.

Also, Protocol Binding—Multi-point WAN must be accomplished as shown in FIG. 30. The following steps must be accomplished.

D1. This event can only occur after event B5 occurs. B5 is the MediaRegistration event which indicates the media for the specified PP is available. This event notifies each protocol which is provisioned under the specified protocol port of the media's availability.

D2. The protocol, upon receipt of this mediaAvailability message can bind to a PP. This binding determines the actions which should occur for packets received for the specified protocol type.

D3. The BindProtocolPort event is used to send the bind information associated with the individual protocols to the appropriate LPs. Included in this bind information is Protocol Forwarding Data and Protocol Specific PP Forwarding Data. Protocol Forwarding Data includes the data specific to each protocol which is required by the forwarding software to forward packets. this information was included in the Protocol Registration (event A6). Also, the Protocol Specific PP Forwarding Data includes forwarding data required by each protocols forwarding software which is associated with an individual PP.

D4. If the Protocol has not yet been enabled on this LP, the appropriate Protocol ForwardingEventHandler is called.

D5. The following events can occur in any order. Update the PP forwarding data in the FaPP. Also, Register with the LCM (Message is to be provided by the LCM).

D6. Update the state of the BTM—Update the forwarding state for the BTM.

At this point, packets received by the Frame Source can be forwarded. The forwarding process works as shown in FIG. 31 (i.e., Packet Forwarding—Multi-point WAN).

El. The dispatcher pulls a packet off the inbound packet queue and calls the appropriate Media frame source with the physical channel number which the packet came from.

E2. The Frame Source gets a pointer to the Inbound PP object via its own mapping from DLCI to PPID and then from PPID to PP object pointer via the logicalPortInfo structure maintained by the FA.

E3. The inbound Media DME is called to process the packet.

E4. The protocol Forwarding DME is determined via the BTM. The inbound PP object has a pointer to the BTM.

E5. The protocol forwarding DME is called. The appropriate protocol processing is done.

E6. The protocol forwarding DME does a cache lookup to determine the outbound PP for which the packet is to be transmitted out. (Part of the cache entry is a pointer to the FaProtocolPort object)

E7. The outbound Media DME is retrieved from the FaProtocolPort object.

E8. The outbound Media DME is called to process the packet.

E9. The appropriate media processing is done and the packet is transmitted out the interface.

Shown in FIG. 32 is the PP initialization for the Point to Point Protocol (PPP) WAN Media (i.e., Media Application Creation and Initialization—PPP WAN). The following steps must be accomplished.

B1. CAS Create and Provisioning messages—The Media process is created by CAS and all provisioning messages are sent to this process.

B2. The ProvDone message from CAS indicates that CAS has finished sending CAS provisioning message to the Media for this CAS session.

B3. The Media application requests the process ID associated with the VR process. A linkage attribute in the Media Application defines the VR which is associated with this media. (The standard CNS registration message type is used)

B4. Some time later, the process ID associated with the request esVR is returned to the Media Application process. This event does not occur until event A5 has occurred. (The standard CNS registration message type is used)

B5. The Media Application registers with the VR. This notifies the VR of the Media's availability and process ID. The bindProtocolPortNotification field is set to ensure the Media Application is notified when the PP is bound to. This value is stored in the VR PP.

Also, Forwarding Data Distribution—PPP WAN must be accomplished as shown in FIG. 33. The following steps must be accomplished.

C1. This event depends on event B5 having occured. The Create Protocol Port event sends the generic protocol port information associated with one or more protocol ports to the relevant set of LPs. If this is the first PP which is created on this LP, PP information for all registered PPs are also sent to this LP. If this is not the first PP which is created on this LP, only the PP information for the registered PP is sent to this LP. The relevant set of LPs is all LPs which have a physical interface in the VR.

C2. Call the Media ForwardingEventHandler—Pass the Media Forwarding Data. The Media FEH creates the Media DME and any other Media required entities.

C3. If the LP which receives the create PP message is in the inboundLpSet, a FainboundProtocolPort object is created, otherwise a FaOutboundProtocolPort object created.

C4. If the LP which receives the create PP message is in the inboundLpSet, create a BindTableManager and a fast BindTableManager and store pointers to then in the FaInboundProtocolPort object.

C5. If the ifEntryRegistration field is set in the faCreateProtocolPort message, space is allocated in LSM for the ifEntry. (LSM provides the message for this event.)

C6. Update the physicalPortInfo Structure—Set the pointer to the FaPP which is associated with the physical channel.

Also, Protocol Binding—PPP WAN must be accomplished as shown in FIG. 34. The following steps must be accomplished.

D1. This event can only occur after event B5 occurs. B5 is the MediaRegistration event which indicates the media for the specified PP is available. This event notifies each protocol which is provisioned under the specified protocol port of the media's availability.

D2. The protocol, upon receipt of this mediaAvailability message can bind to a PP. This binding determines the actions which should occur for packets received for the specified protocol type.

D3. The BindProtocolPort event is used to send the bind information associated with the individual protocols to the appropriate LPs. Included in this bind information is Protocol Forwarding Data and Protocol Specific PP Forwarding Data. The mediaApplicationBindNotification field is set in this message. The Protocol Forwarding Data includes the data specific to each protocol which is required by the forwarding software to forward packets. this information was included in the Protocol Registration (event A6). The Protocol Specific PP Forwarding Data includes forwarding data required by each protocol's forwarding software which is associated with an individual PP.

D4. If the Protocol has not yet been enabled on this LP, the appropriate Protocol ForwardingEventHandler is called.

D5. The following events can occur in any order. Update the PP forwarding data in the FaPP. Also, register with the LCM (Message is to be provided by the LCM).

D6. Update the state of the BTM—Update the forwarding state for the BTM.

D7. A bind notification is sent to the media Application.

At this point, packets received by the Frame Source can be forwarded. The forwarding process works as shown in FIG. 35 (i.e., Packet Forwarding—PPP WAN).

E1. The dispatcher pulls a packet off the inbound packet queue and calls the appropriate Media frame source with the physical channel number which the packet came from.

E2. The Frame Source gets a pointer to the Inbound PP object via its own mapping from DLCI to PPID and then from PPID to PP object pointer via the logicalPortInfo structure maintained by the FA.

E3. The inbound Media DME is called to process the packet.

E4. The protocol Forwarding DME is determined via the BTM. The inbound PP object has a pointer to the BTM.

E5. The protocol forwarding DME is called. The appropriate protocol processing is done.

E6. The protocol forwarding DME does a cache lookup to determine the outbound PP for which the packet is to be transmitted out. (Part of the cache entry is a pointer to the FaProtocolPort object)

E7. The outbound Media DME is retrieved from the FaProtocolPort object.

E8. The outbound Media DME is called to process the packet.

E9. The appropriate media processing is done and the packet is transmitted out the interface.

The Virtual Link is a medium which supports Inter-Virtual Router connectivity. That is the ability to interconnect two Virtual Routers. Although it is possible to do this using a real LAN connection and two physical ports, this mechanism provides the same functionality without the involvement and expense of hardware.

This section describes the VR requirements to support Inter-Virtual Router (IVR) links. Multiple solutions were conceived regarding methods of logically connecting VRs. The solution chosen was to use a point to point model called a "VirtualLink".

Forwarding a packet from one VR through another VR and out a physical interface requires forwarding data (local cache and protocol port information) for both VRs, the inbound VR and the outbound VR. Two alternatives were discussed regarding the location of forwarding data on the VRs. One solution was to put forwarding data on all LPs which could potentially send a packet out a VR. Given this alternative, each LP had to have routing data for VRs which had physical interfaces on its LP as well as VRs which it was connected to via IVR links. This solution was not deemed usable because of the amount of memory it required.

The second solution was to provide only the routing data for VRs supported by this LPs physical interfaces. This would eliminate the need to have routing data for all VRs which could be reached via IVR links. The major disadvantage with this solution is that total packet processing could not be achieved on one LP for IVR packets. If the inbound LP did not have physical interfaces for the outbound VR, it would not have routing data for the outbound VR, thus the packet would have to be sent to an LP which had the data. This requires an additional amount of effort to forward a packet, however the impact of performance for IVR packets doesn't seem to be a major concern.

Virtual Link Data identifies the remote PP and VR associated with a IVR PP. This data exists in the Virtual Link component. Because the Virtual Link component is under casRoot (thus the VR does not get the data for it), it must send this data to both VRs involved and is stored in the appropriate PP objects. This requires a VirtualLinkMedia process to accept the CAS provisioned data and forward it to the VRs.

The second part of the Virtual Link data is data which identifies which LPs have forwarding data for each VR/protocol pair. This data is stored in on each VR in an object called VirtualLinkData. The Virtual Link Data consists of a two dimensional array indexed by LP number and VR number. The data is learned by the VR, and forwarded to all LPs. When this data is received on the LPs, the FA determines the LP to be used for packet processing for each IVR PP on the LP. To determine the LP to be used, a forward search of the array starting with the current LP is done until

an LP is found that supports the destination VR. Using this method, the current LP always is used if possible. (The virtualLink data is forwarded to LPs if one or more PPs are registered with Virtual Link Media)

The Virtual Link Media Application is used to support inter-VR links. The Virtual Link Media initialization process is identical to the process used by the LAN media. However there are some difference in the packet forwarding process. The packet forwarding process for Virtual Link (VL) media is shown in FIGS. 36 and 37. The following steps must be accomplished.

El. The dispatcher pulls a packet off the inbound packet queue and calls the appropriate Media frame source with the physical channel number which the packet came from (This is the standard physical PP, not the IVR PP).

E2. The Frame Source gets a pointer to the Inbound PP object via the physicalPortInfo structure maintained by the FA.

E3. The inbound Media DME is called to process the packet.

E4. The protocol Forwarding DME is determined via the BTM. The inbound PP object has a pointer to the BTM.

E5. The protocol forwarding DME is called. The appropriate protocol processing is done.

E6. The protocol forwarding DME does a cache lookup to determine the outbound PP for which the packet is to be transmitted out. (Part of the cache entry is a pointer to the FaProtocolPort object)

E7. The outbound Media DME is retrieved from the FaProtocolPort object.

E8. The outbound Media DME is called to process the packet. This is the outbound media application DME for the IVR PP.

The IVR media outbound DME determines the far end VR. This information is part of the PP forwarding data.

The packet context is changed, specifically the VR number is changed to the destination VR and the inbound PP number is changed to the IVR PP number.

The IVR media outbound DME determines which LP has routing information for this protocol and this VR. (This information is provided by the VR in the VirtualLinkData table. A forward search is done of the table by VR and LP number. The first LP found which supports this VR and protocol is used.)

E9. The packet is sent to the LP identified in the previous step (if the current LP supports the destination VR, then this step is skipped)

E10. If the packet was forwarded to another LP, the packet is received by the dispatcher, and the Frame Source is called.

E11. The Frame source determines that this is an IVR packet and gets a pointer to the inbound PP associated with the inbound PPID.

E12. The inbound Media Application DME is called (this is the IVR media inbound DME).

E13. The protocol type is determined, and the protocol DME is retrieved from the BTM.

E14. The protocol DME is called to forward the packet.

E15. The protocol DME does the appropriate packet processing including a cache lookup on the destination address in the packet. A successful cache lookup provides a pointer to the outbound PP for which the packet is to be transmitted.

E16. The outbound Media DME is retrieved from the PP object.

E17. The protocol DME fragments the packet if necessary and transmits it out the outbound PP by calling the outbound media DME as identified in the PP structure.

E18. The Media Application outbound DME adds the appropriate media header and sends the packet out the interface.

The Virtual Link simulates the transmission of a packet out an interface of one VR and the reception of the packet on an interface on the destination VR. A packet which is received on an interface is sent to the appropriate forwarder based on the packet type and the bind information associated with the inbound PP. The forwarder, based on routing information, determines the outbound PP is a IVR PP. The packet is transmitted out the PP (via the outbound media DME). It is the responsibility of the IVR PP Media DME to get the packet to a LP which has routing data for the destination VR. This information if provided by the FA in the VirtualLinkData object. If the current LP has the appropriate routing data, the packet is given back to the forwarding software with an updated frame descriptor which includes a new VR ID and new inbound PPID. The forwarding software can then restart the packet forwarding process using the routing data of the outbound VR. Shown in FIG. 38 is an example where the outbound physical port is on the same LP as the inbound physical port.

If the LP which to packet was received on does not have the routing data for the outbound VR, the packet must be transmitted to an LP which does. Again the outbound IVR PP is responsible for getting the packet to an LP which supports the outbound VR. If the current LP does not support the outbound VR, the PP sends the packet to a LP which does. Once the packet is received on the LP which has appropriate routing data, the packet forwarding process is restarted using the new VR ID and incoming PP ID. See an example in FIG. 39. In this example, the outbound PP is an IVR PP which is connected to VR/1. LP1 does not have routing data for VR/1, so the packet must be sent to a LP which does have the routing data, in this case LP2. If the destination LP is not the same as the current LP, the packet could possible traverse three LPs before exiting the platform. This happens if the inbound LP does not support the outbound VR and the packet is sent to another LP for forwarding. On the second LP, the forwarding is done using the destination VR LCM. The outbound PP may be on another LP, which would be the third LP. See this example involving three LPs in FIG. 40.

A ClusterBridge (CB) port can act as a gateway for all bridged ports in the VR to routing and visa versa. The objective is to allow protocol packets to be freely bridged between LAN interfaces, with the freedom of packets destined for the Media Access Control (MAC) Address of the Cluster bridge to be routed outside of the bridge set. The ClusterBridge port is similar to the IVR PP in-that it is a distributed PP. The inbound PP data for CB PP must exist on all LPs supporting the VR. This is required on all LPs with bridging so that packets can be bridged to it and routed out of it. It is required on all LPs with routing so that packets can be routed to it and bridge out of it.

The VR process provides a mechanism which sends inbound and outbound data (FaInboundProtocolPort) to all LPs for distributed PPs. The FA provides this data to the forwarding software upon request.

In order for this to operate successfully with bridge ports, the Cluster Bridge port Bind, Incoming and Outgoing port information is required on every LP which has a bridged port on it per VR.

Cluster Bridge Media is used to support Cluster Bridging. The initialization required prior to packet forwarding is identical to that of the LAN case. (The only special case involved here is that the CB PP is a distributed PP. That is it must exist on all the LPs which contain bridged ports of the same domain. To simplify the PP forwarding data

distribution software, this data is placed on all LPs in the VR.)

However, the process used to forward packet is changed somewhat. This process is identified in FIG. 41 as Packet Forwarding—Cluster Bridge Media. In this example, a packet which is received on a routed port is to be transmitted out a bridged port. The following steps must be accomplished.

El. The dispatcher pulls a packet off the inbound packet queue and calls the appropriate Media frame source with the physical channel number which the packet came from.

E2. The Frame Source gets a pointer to the Inbound PP object via the physicalPortInfo structure maintained by the FA.

E3. The inbound Media DME is called to process the packet.

E4. The protocol Forwarding DME is determined via the BTM. The inbound PP object has a pointer to the BTM.

E5. The protocol forwarding DME is called. The appropriate protocol processing is done. (This is the standard bridging DME) The protocol forwarding DME does a cache lookup to determine the outbound PP for which the packet is to be transmitted out. (There is a static entry in the bridging cache which points to the CB protocol port. At this point, a flag is set in the frameDescriptor which identifies this packet as being originated from bridging)

E6. (Part of the cache entry is a pointer to the FaProtocolPort object)

E7. The outbound Media DME is retrieved from the FaProtocolPort object.

E8. The outbound Media DME is called to process the packet.(This is the ClusterBridge outbound DME).

E9. The CB outbound DME determines if the packet originated from bridging or routing. If it originated from bridging, the protocol type is decoded and the protocol DME is determined via the Bind Table.

E10. The protocol DME is called.

E11. The protocol DME does the appropriate packet processing including a cache lookup on the destination address in the packet. A successful cache lookup provides a pointer to the outbound PP for which the packet is to be transmitted.

E12. The protocol DME fragments the packet if necessary and transmits it out the PP by calling the outbound DME as identified in the PP structure.

E13. The Media Application outbound DME adds the appropriate media header and sends the packet out the interface.

E14. The appropriate media processing is done and the packet is transmitted out the interface.

Beyond initialization and provisioning as described above, it is also desirable to have interface statistics easily retrievable. This can be accomplished in one of several manners known to those of ordinary skill in the art. For example, SNMP may be used as the framework to set up and retrieve this information in a timely fashion.

It will be appreciated by those skilled in the art that the VR system processes including the Virtual Router Process, the Forwarding Agent Process, and the Network Protocol Base Process may each have several components which have specific events that must be acted upon. However in view of the preceding discussion concerning the packet processing and forwarding process which includes virtual routers and virtual links, the software code necessary to accommodate these events in accordance with the present invention should be well within the understanding of a person of ordinary skill in the art. As such no further discussion needs to be provided concerning these events.

Although the invention has been described and illustrated with a certain degree of particularity, it is understood that the present disclosure of embodiments has been made by way of example only and that numerous changes in the arrangement and combination of parts as well as steps may be resorted to by those skilled in the art without departing from the spirit and scope of the invention as claimed.

What is claimed is:

1. A physical switching device for use in a communication network to switch Open Systems Interconnection (OSI) network layer protocol data units within the communication network, the physical switching device comprising:

(a) at least a first and a second virtual switch, each virtual switch comprising decision means for determining an associated directive based on a destination identifier within a particular protocol data unit received at a data port, each virtual switch further comprising processing means for inserting the particular protocol data unit into an outgoing data stream on another data port according to the associated directive to enable delivery of the protocol data unit to the destination identifier, both data ports being associated with a set of data interfaces selected from a plurality of data interfaces in a physical communication network switch, the set of data interfaces being assigned exclusively to a unique virtual switch;

(b) management means, operatively coupled to each virtual switch, for maintaining information on an association between the plurality of data interfaces and each virtual switch, the management means comprising control means dependent on the association information for limiting the processing means of each virtual switch to only inserting the particular protocol data unit into an outgoing data stream on another data port associated with the same virtual switch which received the particular protocol data unit.

2. The physical switching device of claim 1 wherein each data port is selected from the group consisting of protocol data units arriving on a data interface having unique attributes, a data interface on the physical switch, a time slot out of several time slots in a time-divided frame received at a data interface on the physical switch, and a code divided cell out of several code divided cells received at at least one data interface on the physical switch.

3. The physical switching device of claim 1 wherein the set of data interfaces associated with a virtual switch includes a first data interface including means for manipulating a protocol data unit having a different protocol type from a second data interface such that protocol data units of different protocol types can be switched within a single virtual switch, the different protocol data unit protocol types being selected from the group consisting of different Open Systems Interconnection (OSI) physical layer media types, different OSI link layer signaling protocols, and different OSI network layer protocols.

4. The physical switching device of claim 1 wherein the management means further comprises means for maintaining a database of known destination identifiers and means for requiring verification that the destination identifier in the particular protocol data unit is in the database prior to inserting the particular protocol data unit into an outgoing data stream on another data port such that delivery of the protocol data unit to an unknown destination identifier is prevented.

5. The physical switching device of claim 1 wherein each virtual switch processing means comprises means for restructuring the particular protocol data unit by deleting,

inserting, and replacing bits in the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

6. The physical switching device of claim 1 wherein each virtual switch processing means comprises means for monitoring the particular protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

7. The physical switching device of claim 1 wherein:

(a) the physical switching device further comprises means for performing operations selected from the group consisting of bridge, route, switch, in-line filter, protocol conversion, and a security function;

(b) the protocol data unit is selected from the group consisting of a frame, a cell, and a packet;

(c) the communication network is selected from the group consisting of local area network, wide area network, metropolitan area network, and wireless network; and

(d) the communication network switches protocol data units having a content selected from the group consisting of voice, video, and data.

8. A physical switching device for use in a communication network to switch Open Systems Interconnection (OSI) network layer protocol data units within the communication network on a shared communication medium, the physical switching device comprising:

(a) at least a first and a second virtual switch, each virtual switch comprising decision means for determining an associated directive based on a destination identifier within a particular protocol data unit received at a data port, each virtual switch further comprising processing means for inserting the particular protocol data unit into an outgoing data stream on another data port according to the associated directive to enable delivery of the protocol data unit to the destination identifier, both data ports being associated with a set of data interfaces selected from a plurality of data interfaces in a physical communication network switch, the set of data interfaces being assigned exclusively to a unique virtual switch;

(b) virtual link management means, operatively coupled to at least the first and the second virtual switches, for maintaining information on at least one virtual link between at least the first and the second virtual switch, each virtual link comprising a first end and a second end of a data path on the shared communication medium, each virtual link end comprising a data port from the plurality of data interfaces in the physical communication network switch.

9. The physical switching device of claim 8 wherein the first and the second virtual link end of the at least one virtual link are in a different set of data ports assigned exclusively to the first and the second virtual switch, respectively, such that the virtual link provides a data path between the first and the second virtual switch on the shared communication medium.

10. The physical switching device of claim 9 wherein the first and the second virtual switches are located in a single geographic location and the shared communication medium comprises a memory shared between the first and the second virtual switches.

11. The physical switching device of claim 8 wherein the first and the second virtual switches are geographically remote from one another and wherein the first and the second virtual link ends are in a single set of data ports assigned exclusively to the first and the second virtual switch such that the virtual link provides a data path between the first and the second virtual switches on the shared communication medium across a geographic distance.

12. The physical switching device of claim 11 wherein the shared communication medium comprises a high data transfer rate link between the first and the second virtual switches which spans the geographic distance.

13. The physical switching device of claim 8 further comprising at least one filter operatively coupled to the data path which filters protocol data units communicated in the virtual link data path according to one access policy out of a plurality of access policies that are separately specified for each virtual switch.

14. The physical switching device of claim 13 wherein the at least one filter comprises a first and a second filter operatively coupled to the first and the second virtual switch, respectively, which filters protocol data units communicated in the virtual link data path according to an access policy specified for the first and the second virtual switch, respectively.

15. The physical switching device of claim 8 wherein each data port is selected from the group consisting of protocol data units arriving on a data interface having unique attributes, a data interface on the physical switch, a time slot out of several time slots in a time-divided frame received at a data interface on the physical switch, and a code divided cell out of several code divided cells received on at least one data interface on the physical switch.

16. The physical switching device of claim 8 wherein the set of data interfaces associated with a virtual switch includes a first data interface including means for manipulating a protocol data unit having a different protocol type from a second data interface such that protocol data units of different protocol types can be switched within a single virtual switch, the different protocol data unit protocol types being selected from the group consisting of different Open Systems Interconnection (OSI) physical layer media types, different OSI link layer signaling protocols, and different OSI network layer protocols.

17. The physical switching device of claim 8 wherein each virtual switch processing means comprises means for restructuring the particular protocol data unit by deleting, inserting, and replacing bits in the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

18. The physical switching device of claim 8 wherein each virtual switch processing means comprises means for monitoring the particular protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

19. The physical switching device of claim 8 wherein:

(a) the physical switching device further comprises means for performing operations selected from the group consisting of bridge, route, switch, in-line filter, protocol conversion, and a security function;

(b) the protocol data unit is selected from the group consisting of a frame, a cell, and a packet;

(c) the communication network is selected from the group consisting of local area network, wide area network, metropolitan area network, and wireless network; and

(d) the communication network switches protocol data units having a content selected from the group consisting of voice, video, and data.

20. A communication system which delivers Open Systems Interconnection (OSI) network layer protocol data units within a first and a second virtual closed user group on a shared communication medium, the communication system comprising:

(a) first virtual closed user group processing means for examining and modifying data bits within a protocol data unit received from a member of the first virtual closed user group on the shared communication medium, each member of the first virtual closed user group having a unique destination identifier, the first virtual closed user group processing means comprising delivery means for delivering the modified protocol data unit to another member of the first virtual closed user group;

(b) second virtual closed user group processing means for examining and modifying data bits within a protocol data unit received from a member of the second virtual closed user group on the shared communication medium, each member of the second virtual closed user group having a unique destination identifier, the second virtual closed user group processing means comprising delivery means for delivering the modified protocol data unit to another member of the second virtual closed user group; and

(c) a framer means, operatively coupled to the first and the second virtual closed user group processing means, for maintaining a database of all destination identifiers representing users in that user group currently reachable for delivery of protocol data units within the communication system, the framer means comprising means for requiring verification that each destination identifier in a protocol data unit indicates a user in that user group can be currently reached for delivery through a lookup in the database prior to completing delivery of the protocol data unit to the user indicated by the associated destination identifier, the framer means further comprising means for limiting access to the database such that each virtual closed user group only has access to specific destination identifiers owned by that particular virtual closed user group so that a protocol data unit having a destination identifier which is not owned by the particular virtual closed user group will not be delivered.

21. The communication system of claim 20 wherein each virtual closed user group processing means comprises means for modifying data bits within a received protocol data unit by deleting, inserting, and replacing bits in the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group.

22. The communication system of claim 20 wherein each virtual closed user group processing means further comprises means for monitoring the received protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual dosed user group.

23. The communication system of claim 20 wherein each virtual closed user group processing means delivers the modified protocol data unit to another member of the same virtual closed user group without modifying predetermined Open Systems Interconnection (OSI) physical layer, link layer, and network layer access protocols used to communicate protocol data units over the shared communication medium.

24. The communication system of claim 23 wherein each virtual closed user group processing means delivers the modified protocol data unit to another member of the same virtual closed user group without modifying the predetermined access protocols such that any particular device capable of communicating on the shared communication medium can be a member of either virtual closed user group by having the framer means limit database access to destination identifiers associated with the particular device to a particular desired virtual closed user group.

25. The communication system of claim 20 wherein the framer means further comprises means for assigning incoming protocol data unit traffic to each virtual closed user group based on an access policy that is separately specified in each virtual closed user group.

26. The communication system of claim 20 wherein the first and the second virtual closed user group processing means include a first and a second virtual switch, respectively, each virtual switch comprising decision means for determining an associated directive based on a destination identifier within a particular protocol data unit received at a data port, each virtual switch further comprising a processor, which performs the functions of the virtual closed user group delivery means by inserting the particular protocol data unit into an outgoing data stream on another data port according to the associated directive to enable delivery of the protocol data unit to the destination identifier within the protocol data unit.

27. The communication system of claim 26 wherein the first and the second virtual switch are located within a single physical switching device, both data ports for each virtual switch being associated with a set of data interfaces in the physical switching device assigned exclusively to the same virtual switch.

28. The communication system of claim 26 wherein the first and the second virtual switch are located within different physical switching devices, both data ports for each virtual switch being associated with a set of data interfaces in the respective physical switching devices which are assigned exclusively to the same virtual switch.

29. The communication system of claim 28 wherein the different physical switching devices are geographically remote from one another.

30. The communication system of claim 26 wherein each data port is selected from the group consisting of protocol data units arriving on a data interface having unique attributes, a data interface on a physical switching device, a time slot out of several time slots in a time-divided frame received at a data interface on a physical switching device, and a code divided cell out of several code divided cells received on at least one data interface on a physical switching device.

31. The communication system of claim 26 wherein the set of data interfaces associated with a virtual switch from a physical switching device includes a first data interface including means for manipulating a protocol data unit having a different protocol type from a second data interface such that protocol data units of different protocol types can be switched within a single virtual switch, the different protocol data unit protocol types being selected from the group consisting of different Open Systems Interconnection (OSI) physical layer media types, different OSI link layer signaling protocols, and different OSI network layer protocols.

32. The communication system of claim 26 wherein each virtual switch processor comprises means for restructuring the particular protocol data unit by deleting, inserting, and replacing bits in the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

33. The communication system of claim 26 wherein each virtual switch processor comprises means for monitoring the particular protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

34. The communication system of claim 26 wherein:

(a) the protocol data unit is selected from the group consisting of a frame, a cell, and a packet;

(b) the communication network is selected from the group consisting of local area network, wide area network, metropolitan area network, and wireless network; and

(c) the communication network switches protocol data units having a content selected from the group consisting of voice, video, and data.

35. The communication system of claim 26 further comprising a virtual link between the first and the second virtual switch, the virtual link comprising a first end and a second end of a data path on the shared communication medium, each end comprising a data port in a different virtual closed user group.

36. The communication system of claim 35 further comprising a filter operatively coupled to the data path which filters protocol data units communicated in the data path.

37. The communication system of claim 20 wherein the first virtual closed user group processing means include a first and a second virtual switch, respectively, each virtual switch comprising decision means for determining an associated directive based on a destination identifier within a particular protocol data unit received at a data port, each virtual switch further comprising a processor, which performs the functions of the virtual closed user group delivery means by inserting the particular protocol data unit into an outgoing data stream on another data port according to the associated directive to enable delivery of the protocol data unit to the destination identifier within the protocol data unit.

38. The communication system of claim 37 wherein the first and the second virtual switch are located within a single physical switching device, both data ports for each virtual switch being associated with a set of data interfaces in the physical switching device assigned exclusively to the same virtual switch.

39. The communication system of claim 37 wherein the first and the second virtual switch are located within different physical switching devices, both data ports for each virtual switch being associated with a set of data interfaces in the respective physical switching devices which are assigned exclusively to the same virtual switch.

40. The communication system of claim 37 wherein the different physical switching devices are geographically remote from one another.

41. The communication system of claim 37 wherein each data port is selected from the group consisting of protocol data units arriving on a data interface having unique attributes, a data interface on a physical switching device, a time slot out of several time slots in a time-divided frame received at a data interface on a physical switching device, and a code divided cell out of several code divided cells received on at least one data interface on a physical switching device.

42. The communication system of claim 37 wherein the set of data interfaces associated with a virtual switch from a physical switching device includes a first data interface including means for manipulating a protocol data unit having a different protocol type from a second data interface such that protocol data units of different protocol types can be switched within a single virtual switch, the different protocol data unit protocol types being selected from the group consisting of different Open Systems Interconnection (OSI) physical layer media types, different OSI link layer signaling protocols, and different OSI network layer protocols.

43. The communication system of claim 37 wherein each virtual switch processor comprises means for restructuring the particular protocol data unit by deleting, inserting, and replacing bits in the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

44. The communication system of claim 37 wherein each virtual switch processor comprises means for monitoring the particular protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the particular protocol data unit in accordance with the associated directive prior to inserting the particular protocol data unit into the outgoing data stream.

45. The communication system of claim 37 wherein:

(a) the protocol data unit is selected from the group consisting of a frame, a cell, and a packet;

(b) the communication network is selected from the group consisting of local area network, wide area network, metropolitan area network, and wireless network; and

(c) the communication network switches protocol data units having a content selected from the group consisting of voice, video, and data.

46. The communication system of claim 37 further comprising a virtual link between the first and the second virtual switch, the virtual link comprising a first end and a second end of a data path on the shared communication medium, each end comprising a data port in the same virtual closed user group.

47. The communication system of claim 46 further comprising a filter operatively coupled to the data path which filters protocol data units communicated in the data path.

48. A method for delivering Open Systems Interconnection (OSI) network layer protocol data units within a first and a second virtual closed user group on a shared communication medium in a communication system, the method comprising the device-implemented steps of:

(a) examining and modifying data bits within a protocol data unit received from a member of the first virtual closed user group on the shared communication medium, each member of the first virtual closed user group having a unique destination identifier;

(b) examining and modifying data bits within a protocol data unit received from a member of the second virtual closed user group on the shared communication medium, each member of the second virtual closed user group having a unique destination identifier;

(c) maintaining a database of all destination identifiers representing members which are currently reachable for delivery of protocol data units within the communication system;

(d) limiting access to the database such that each virtual closed user group only has access to specific destination identifiers owned by that particular virtual closed user group;

(e) requiring verification that each destination identifier in a protocol data unit indicates a member which is currently reachable for delivery through a lookup in the database prior to completing delivery of the protocol data unit to the member represented by the associated destination identifier;

(f) delivering the first virtual closed user group modified protocol data unit to another member of the first virtual closed user group after verifying that the first virtual closed user group member destination identifier is currently reachable; and

(g) delivering the second virtual closed user group modified protocol data unit to another member of the second virtual closed user group after verifying that the second virtual closed user group member destination identifier is currently reachable, step (f) and (g) being device-implemented such that a protocol data unit having a destination identifier which is not owned by the particular virtual closed user group will not be delivered.

49. The method of claim **48** wherein each examining and modifying step comprises modifying data bits within a received protocol data unit by deleting, inserting, and replacing bits in the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group.

50. The method of claim **48** wherein each examining and modifying step comprises monitoring the received protocol data unit by dropping, sending, sending a copy of, and auditing the contents of the received protocol data unit prior to delivering the modified protocol data unit to another member of the same virtual closed user group.

51. The method of claim **48** wherein steps (a) through (g) are performed such that all predetermined physical layer, link layer, and network layer access protocols used to communicate protocol data units over the shared communication medium are preserved.

52. The method of claim **51** wherein steps (a) through (g) are performed such that all of the predetermined access protocols are preserved so that any particular device capable of communicating on the shared communication medium can be a member of either virtual closed user group by performing an additional step of adding a destination identifier associated with the particular device to the database.

53. The method of claim **48** further comprising a step of assigning incoming protocol data unit to each virtual closed user group based on an access policy that is separately specified in each virtual closed user group.

54. The method of claim **48** further comprising a step of providing a virtual link between the first and the second virtual closed user group, the virtual link comprising a first end and a second end of a data path on the shared communication medium, each end comprising a data port in a different virtual closed user group.

55. The method of claim **54** wherein the providing step comprises utilizing a shared memory as the shared communication medium to provide the virtual link.

56. The method of claim **54** wherein the providing step comprises utilizing a high data transfer rate link which spans a geographic distance between the first and the second virtual closed user group to provide the virtual link.

57. The method of claim **55** further comprising a step of filtering protocol data units communicated in the virtual link data path according to an access policy.

* * * * *

US 20020165978A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2002/0165978 A1

Chui (43) Pub. Date: Nov. 7, 2002

(54) MULTI-SERVICE OPTICAL INFINIBAND ROUTER

(76) Inventor: Terence Chui, Milpitas, CA (US)

Correspondence Address:
TERENCE CHUI
672 PRINCESS PLACE
MILPITAS, CA 95035 (US)

(21) Appl. No.: 10/139,715

(22) Filed: May 6, 2002

Related U.S. Application Data

(60) Provisional application No. 60/289,274, filed on May 7, 2001.

Publication Classification

(51) Int. Cl.[7] G06F 15/16; G06F 15/173

(52) U.S. Cl. 709/238; 709/230

(57) ABSTRACT

This invention pertains a system and method for interconnecting processing module within a computer device and the input/output channels external to the computer devices. More specifically, the Multi-Service Optical InfiniBand Router (OIR) relates to the use of a device to communicate with InfiniBand devices, IP-based switching devices, IP-based routing devices, SONET Add-Drop Multiplexing devices, DWDM (Dense Wavelength Division Multiplexing) devices, Fibre Channel devices, and SCSI devices.

InfiniBand System Architecture

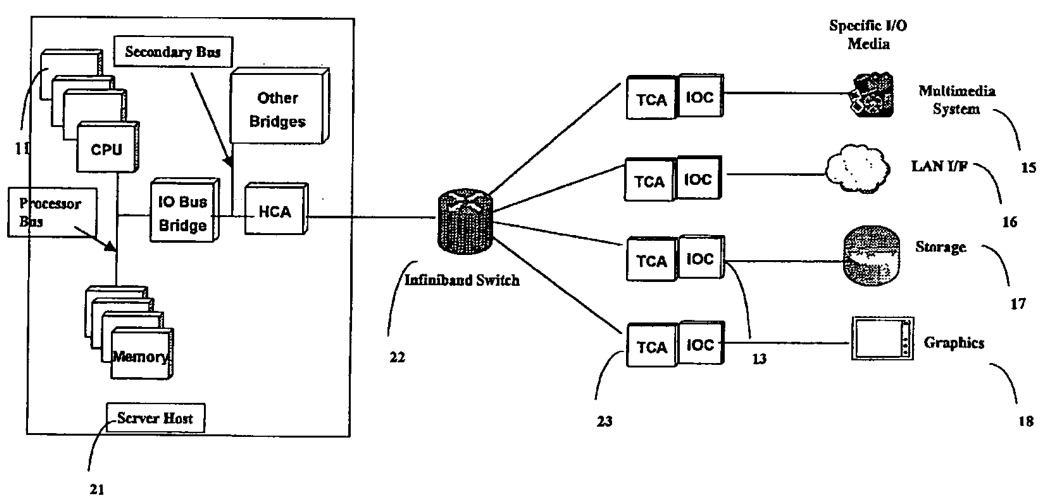


Figure 1. Traditional System Architecture

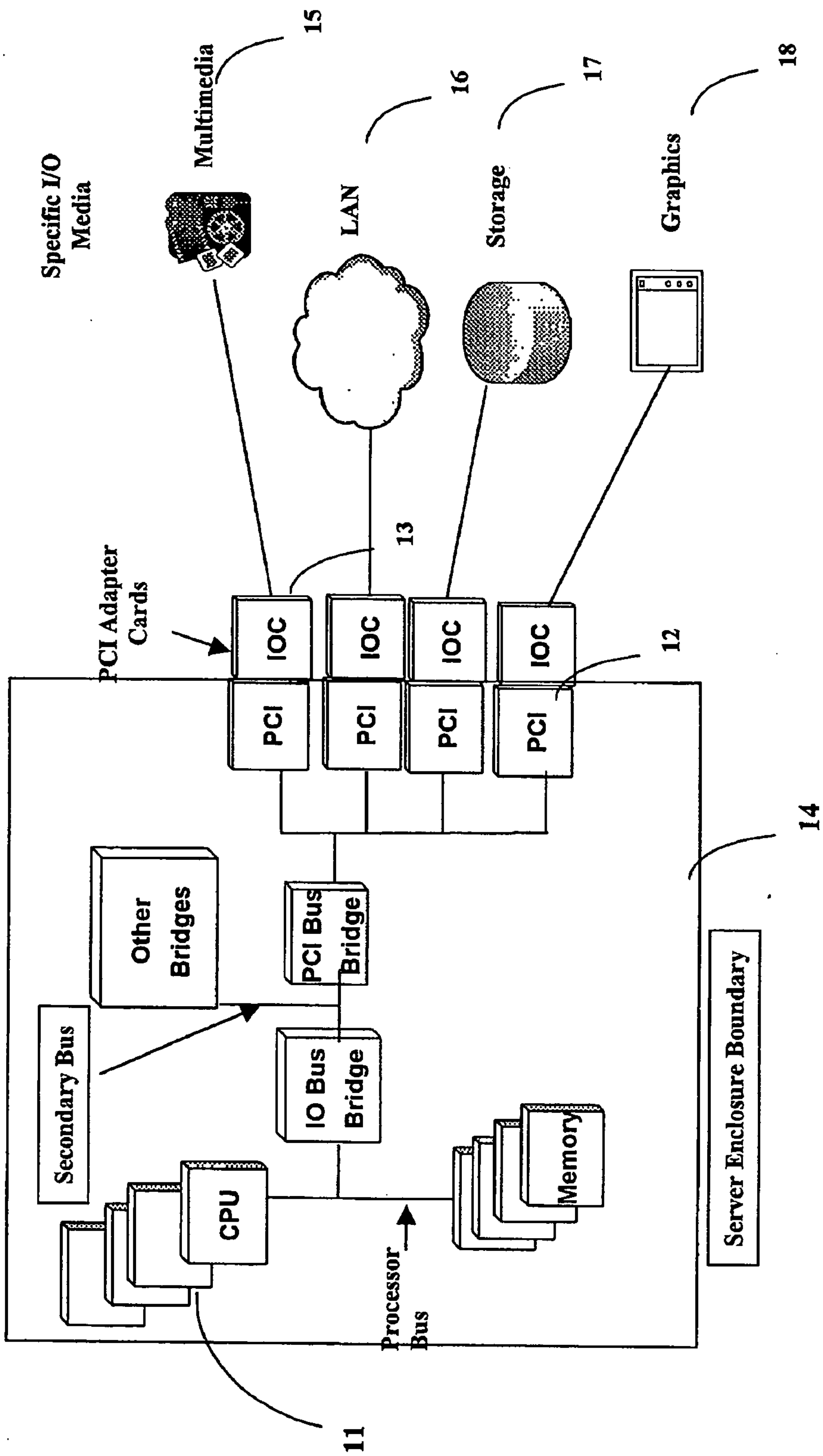


Figure 2. InfiniBand System Architecture

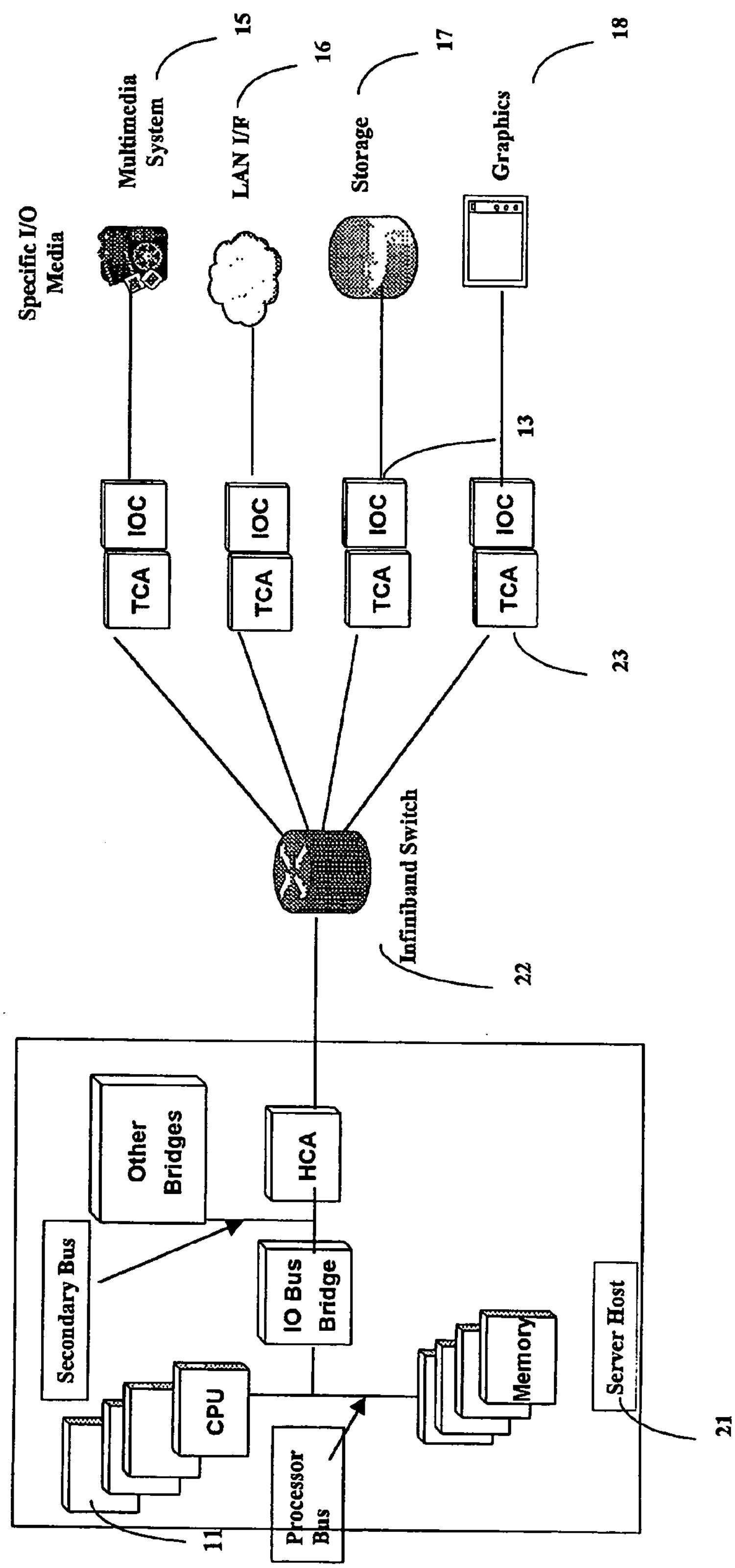


Figure 3. Optical InfiniBand Architecture

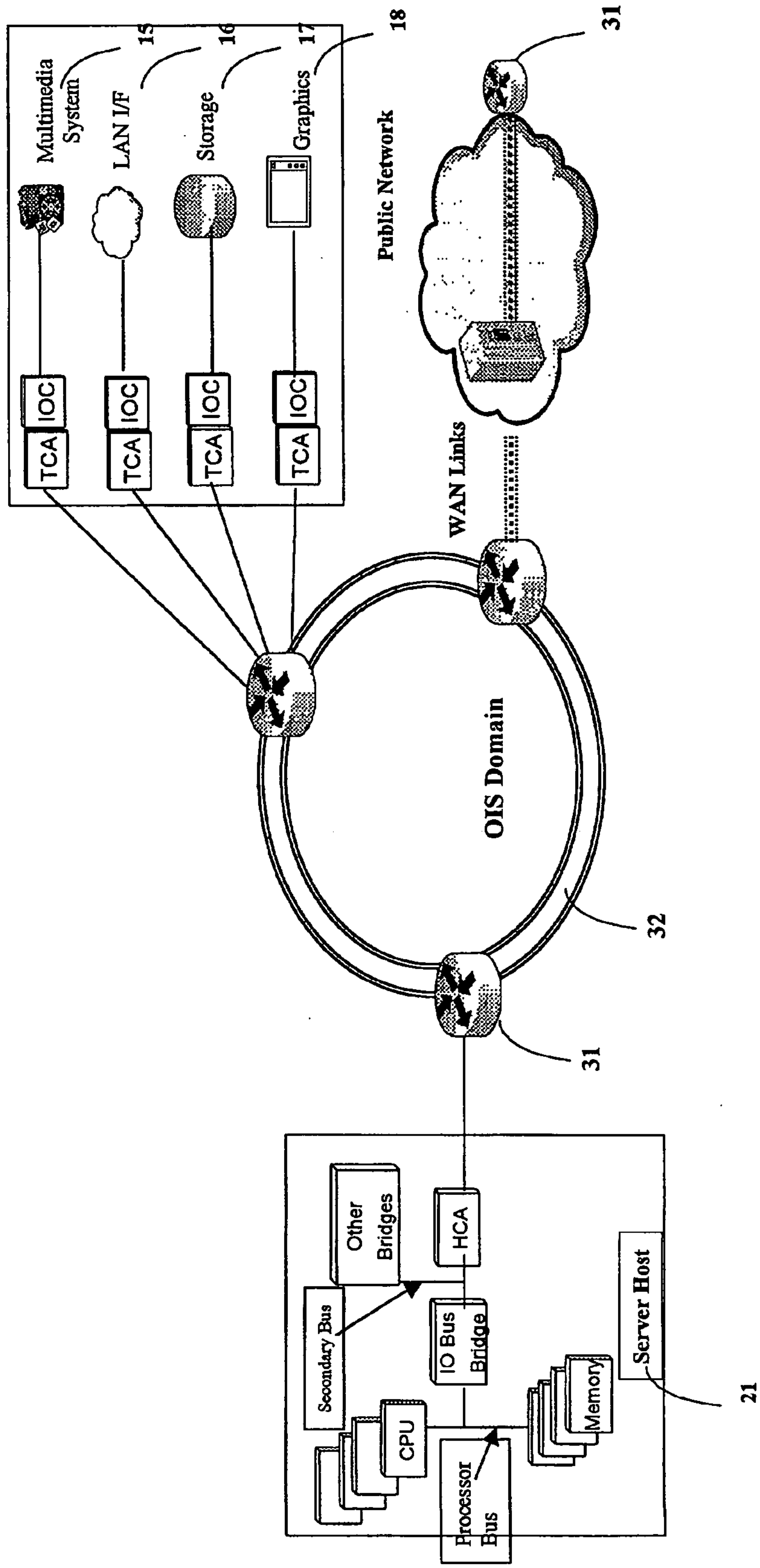


Figure 4. OIR Sample Physical System Layout

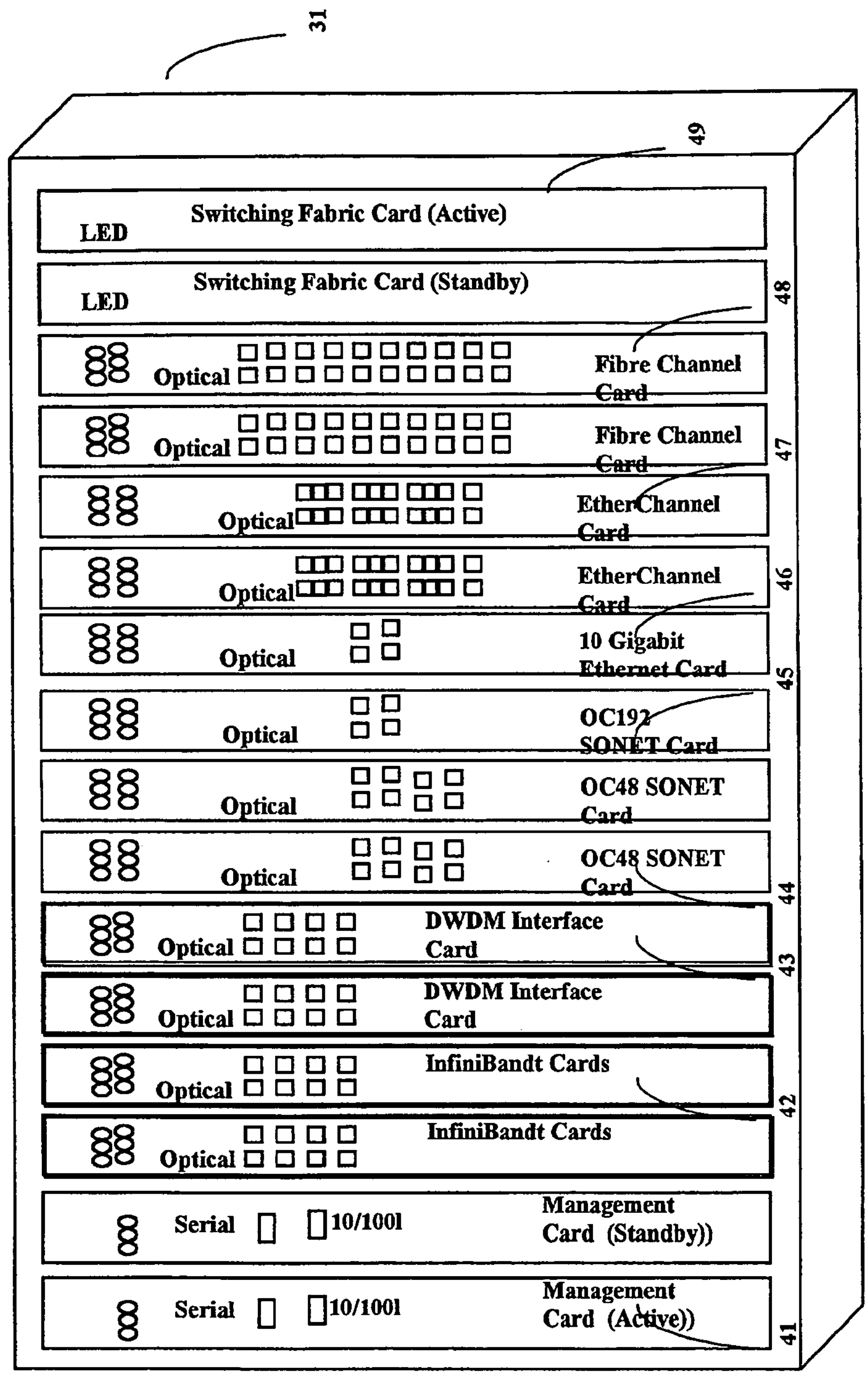


Figure 5. OIR Logical Multi-Services System Layout

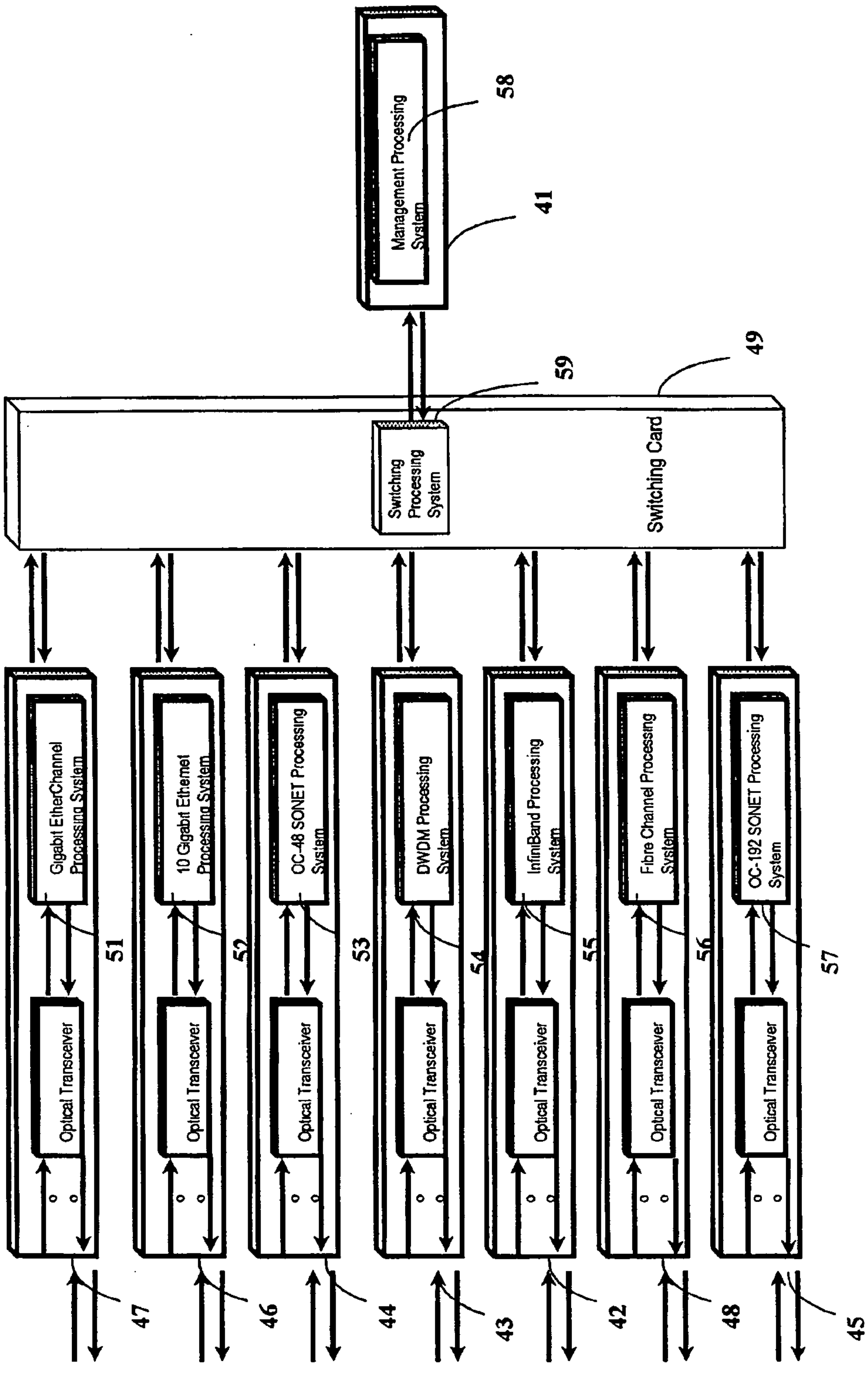


Figure 6. Internetwork SAN Architecture

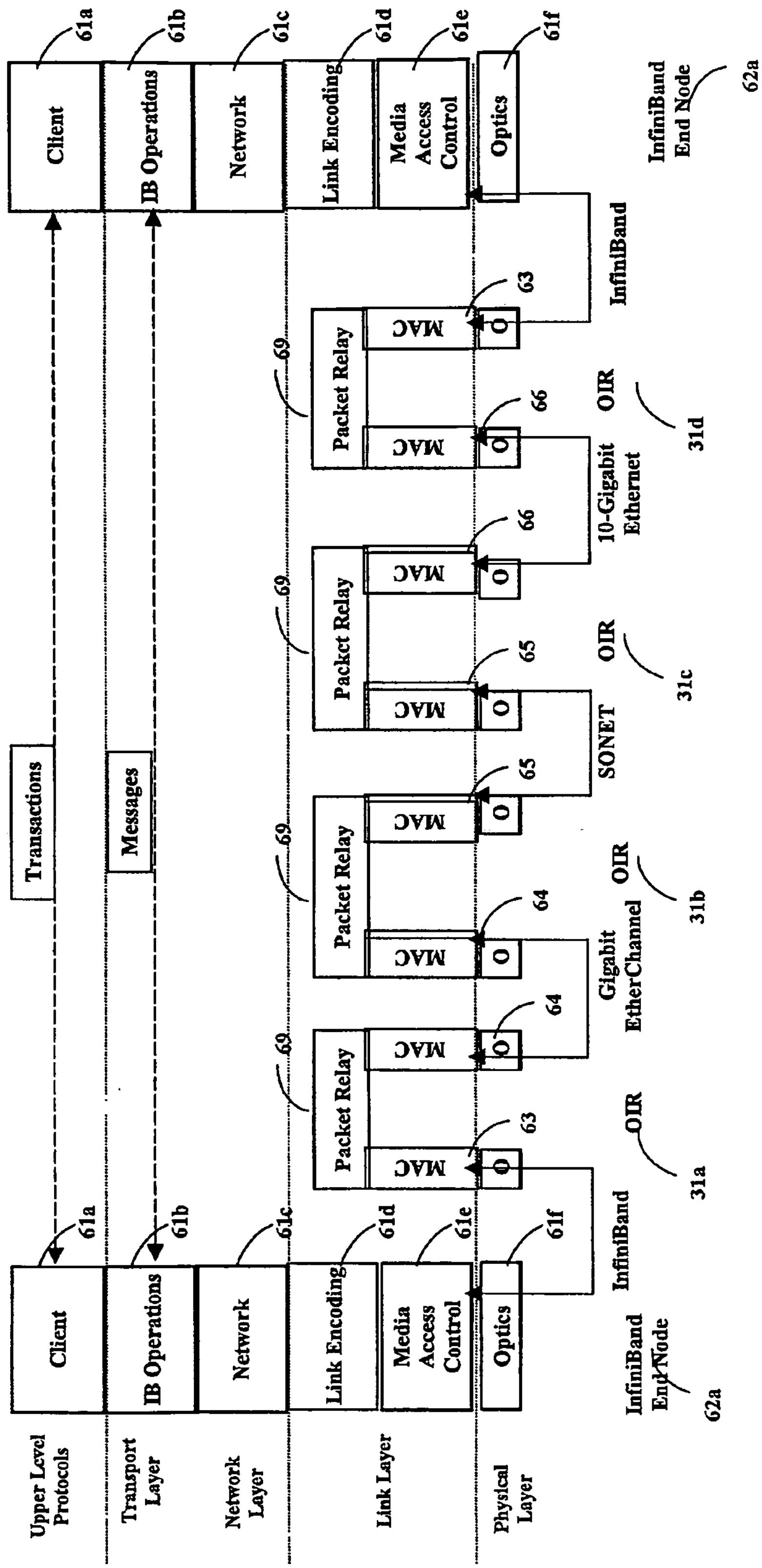


Figure 7: InfiniBand Packet Switching through OIR system

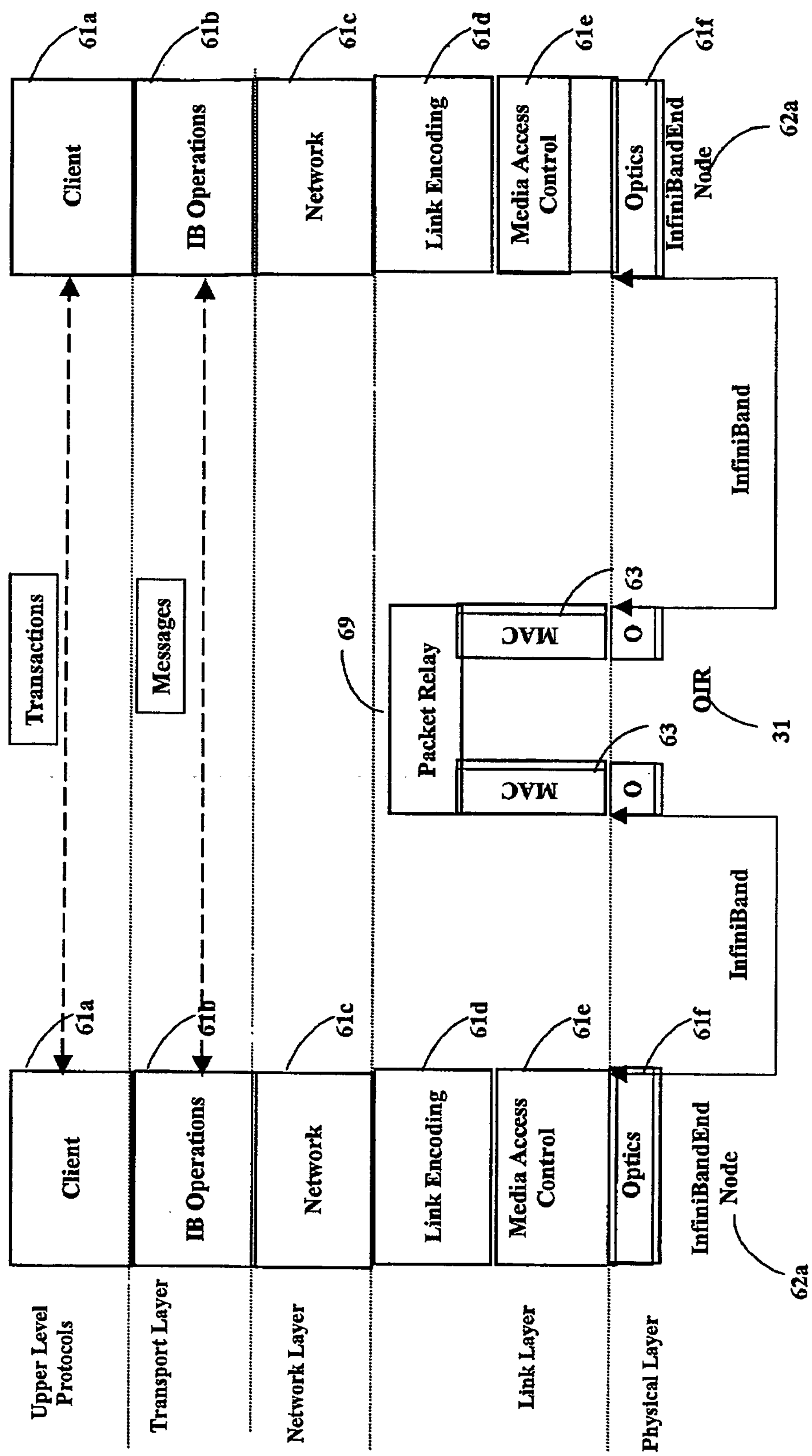
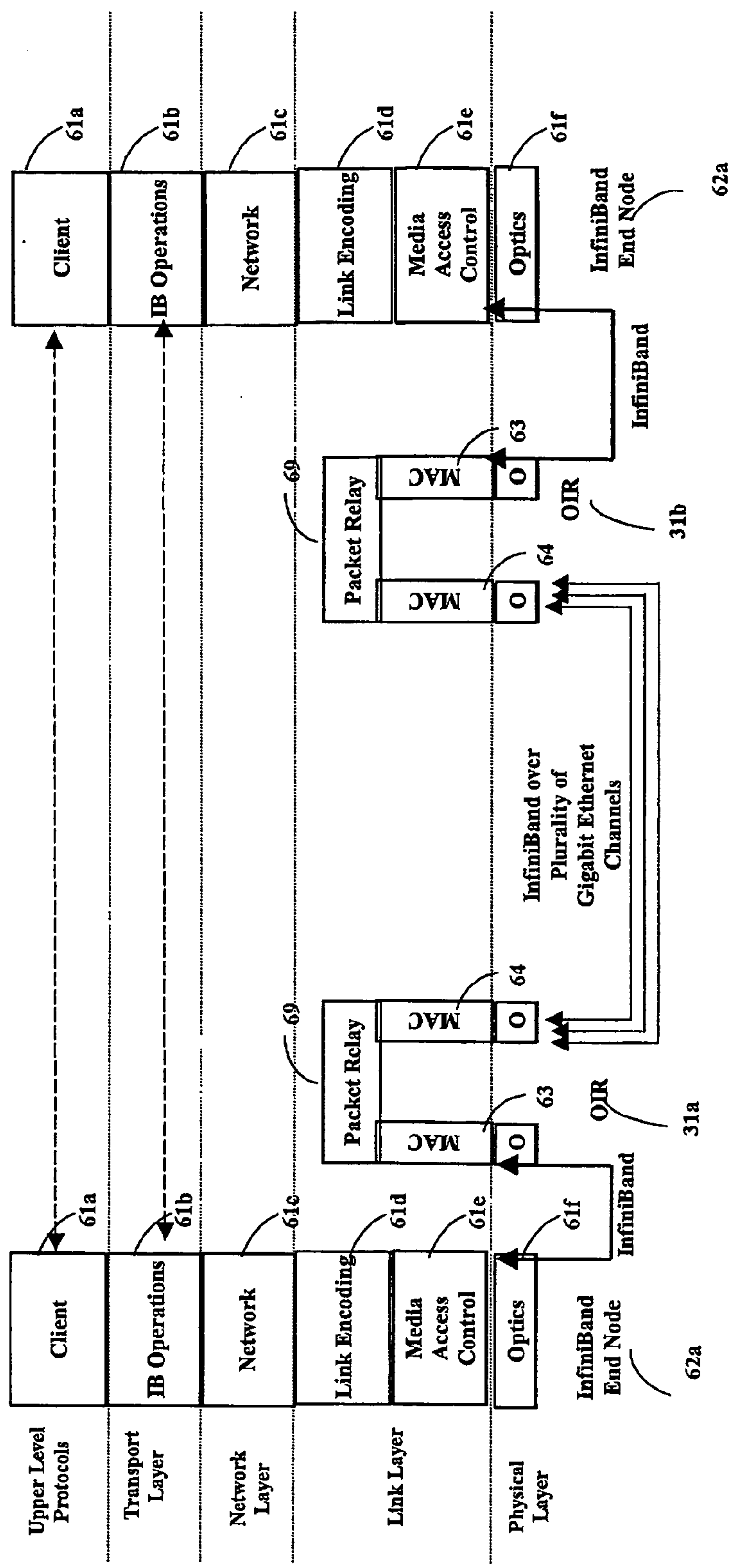


Figure 8. Inter-OIR InfiniBand Packet Switching using Gigabit Ethernet Interfaces



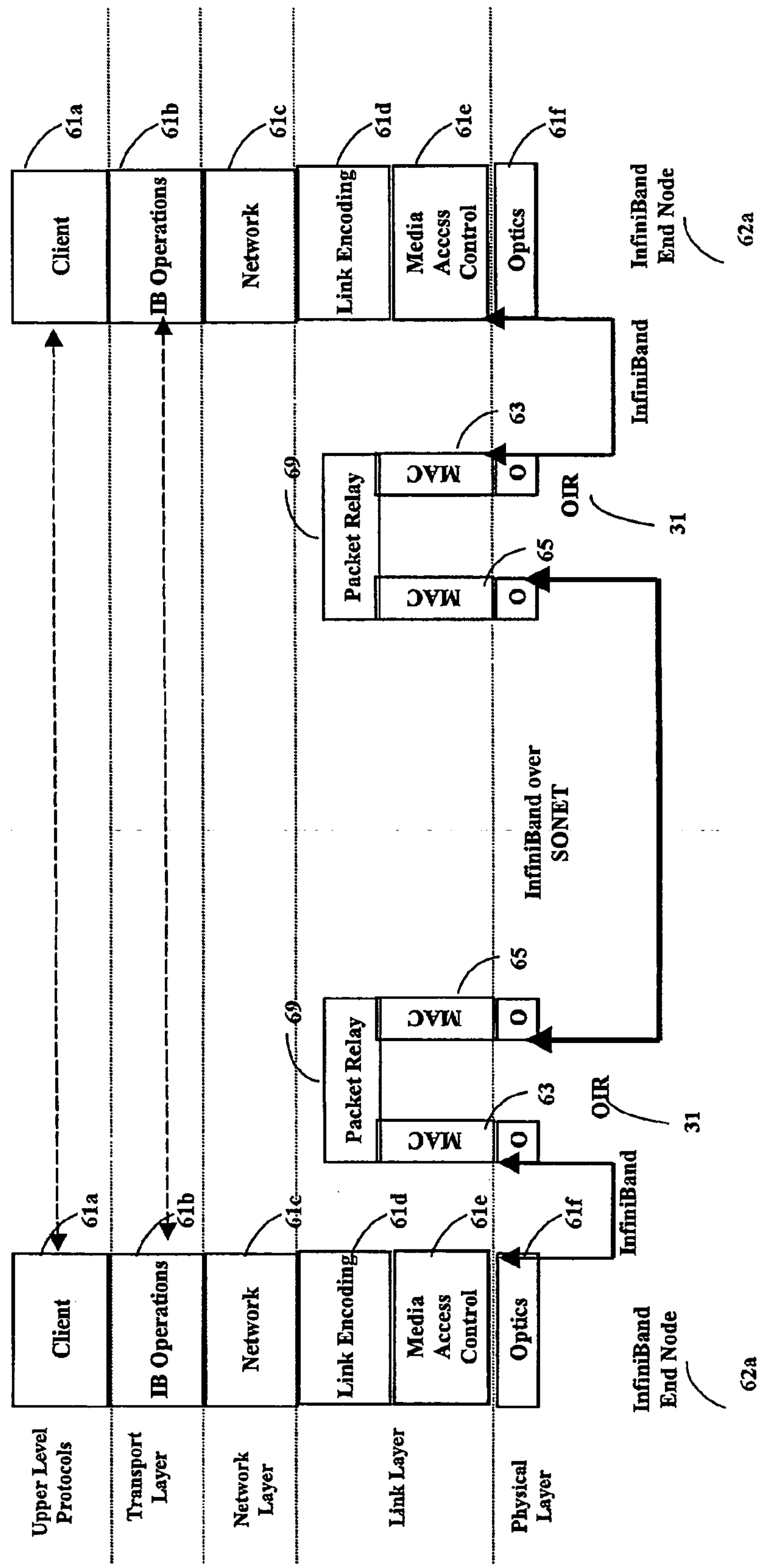


Figure 9. Inter-OIR InfiniBand Packet Switching using SONET Interfaces

Figure 10. Inter-OIR InfiniBand Frames Switching using DWDM Interfaces

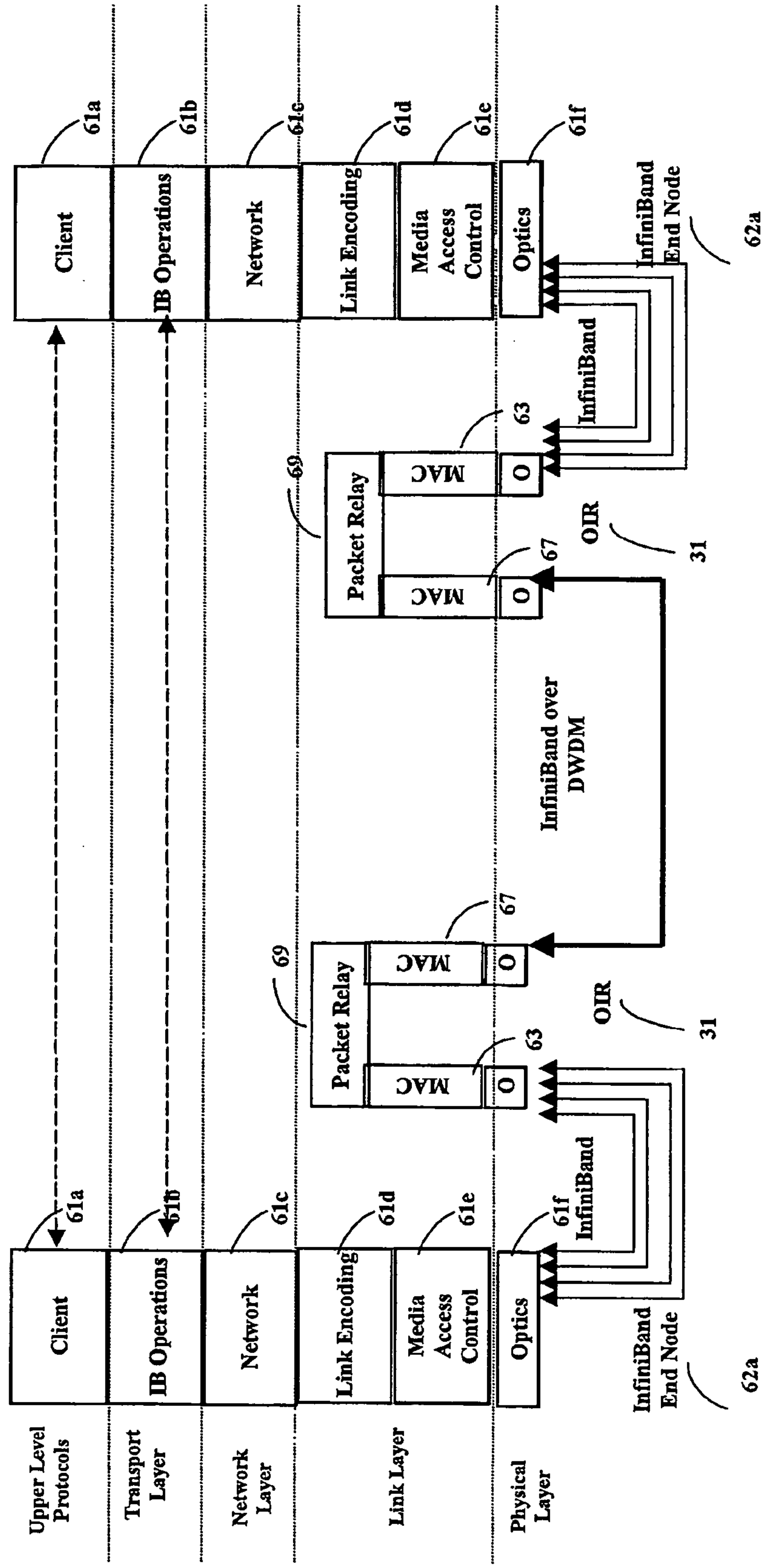
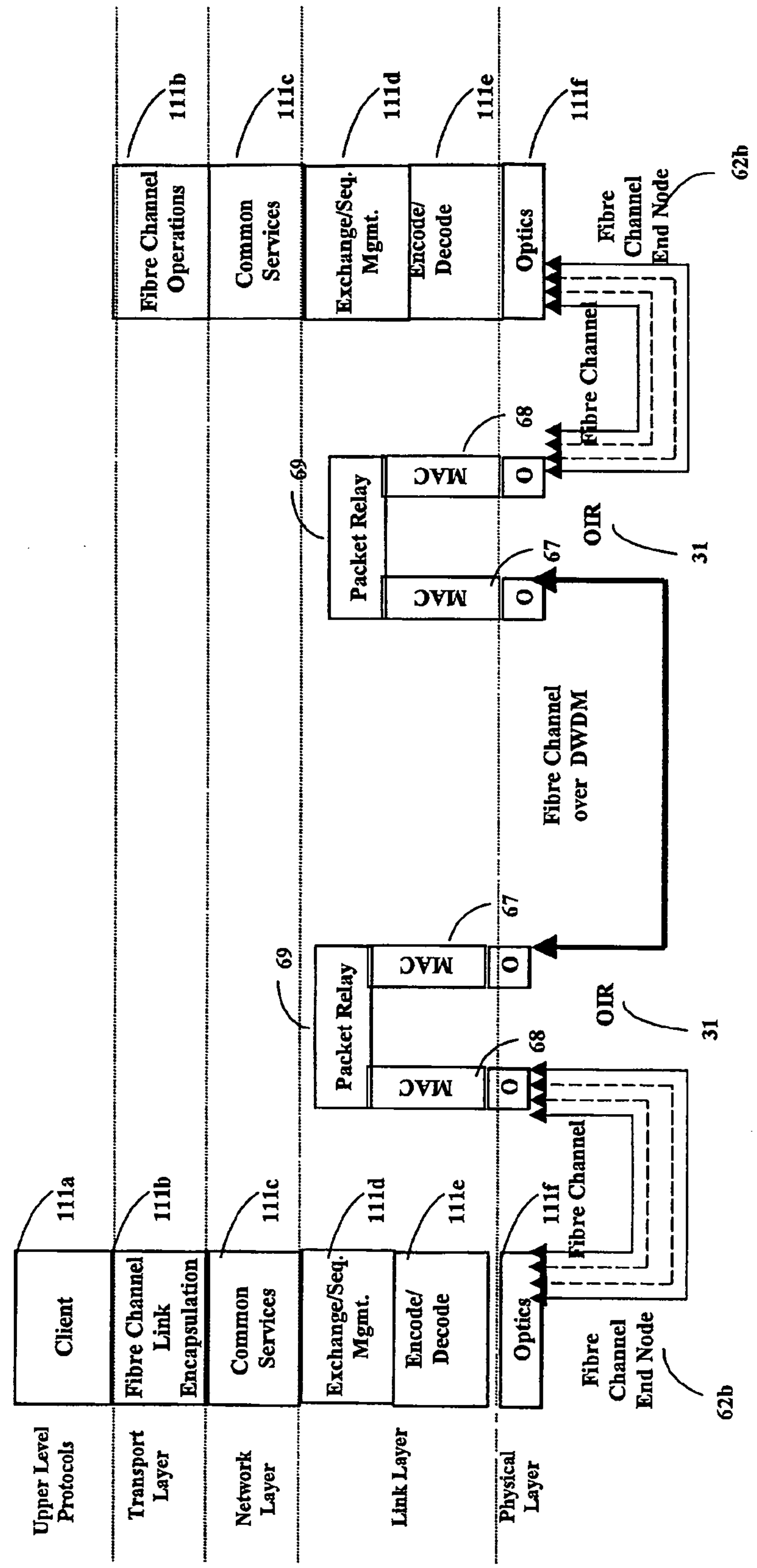


Figure 11. Inter-OIR Fibre Channel Frames Switching using DWDM Interfaces



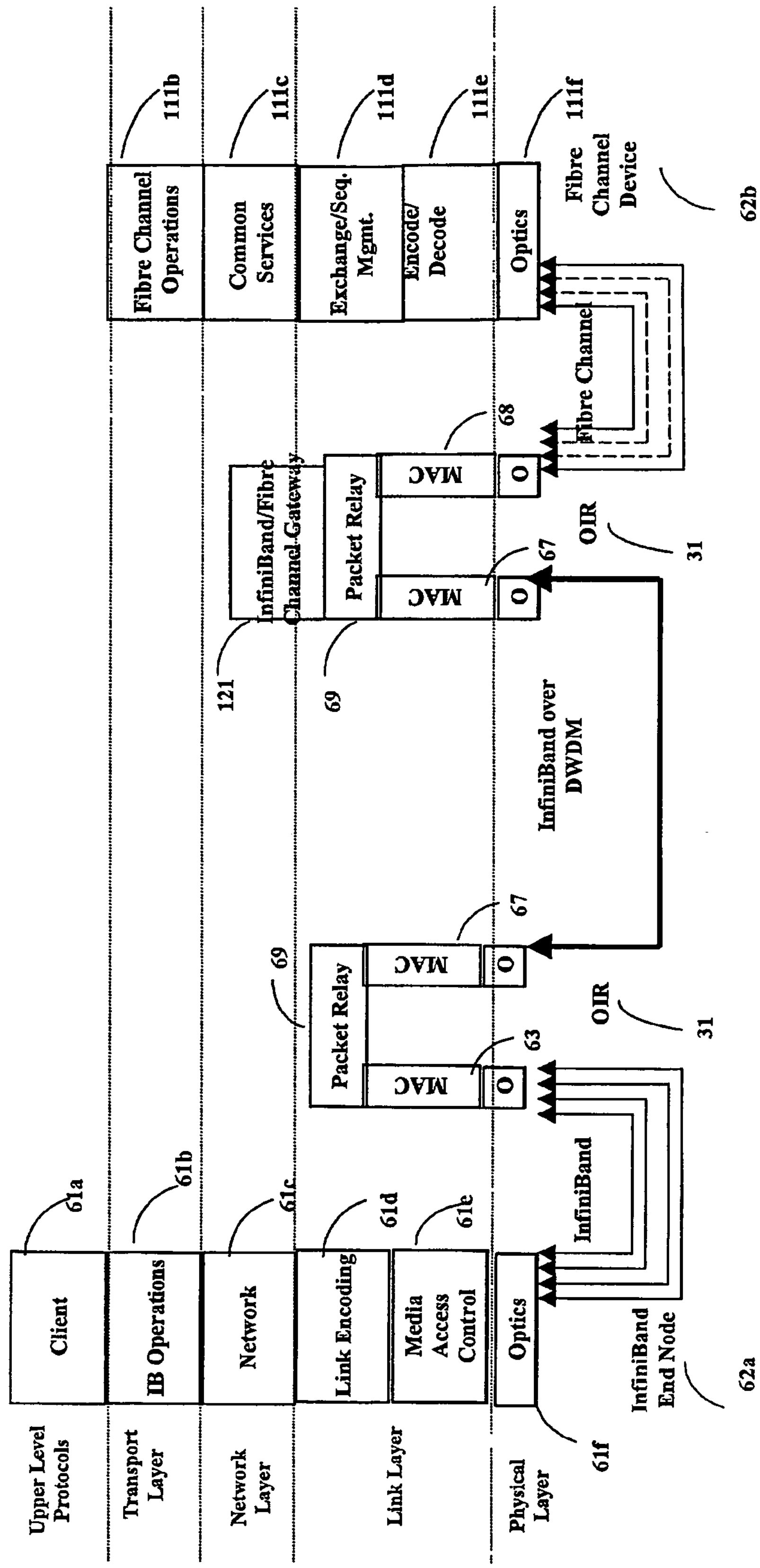


Figure 12. Inter-OIR InfiniBand/Fibre Channel Data Switching using DWDM Interfaces

Figure 13. Inter-OIR InfiniBand/iSCSI Data Switching using DWDM Interfaces

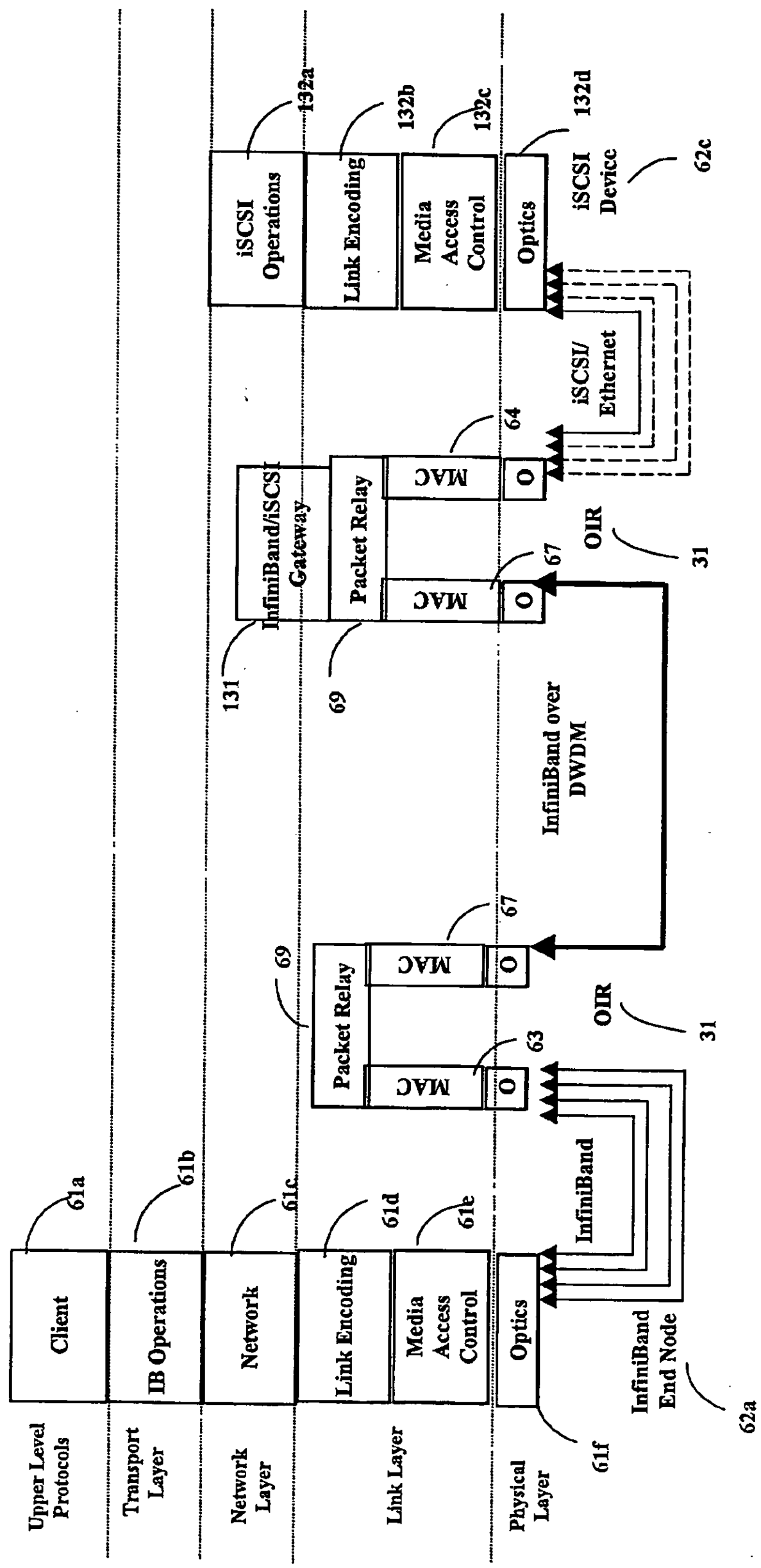
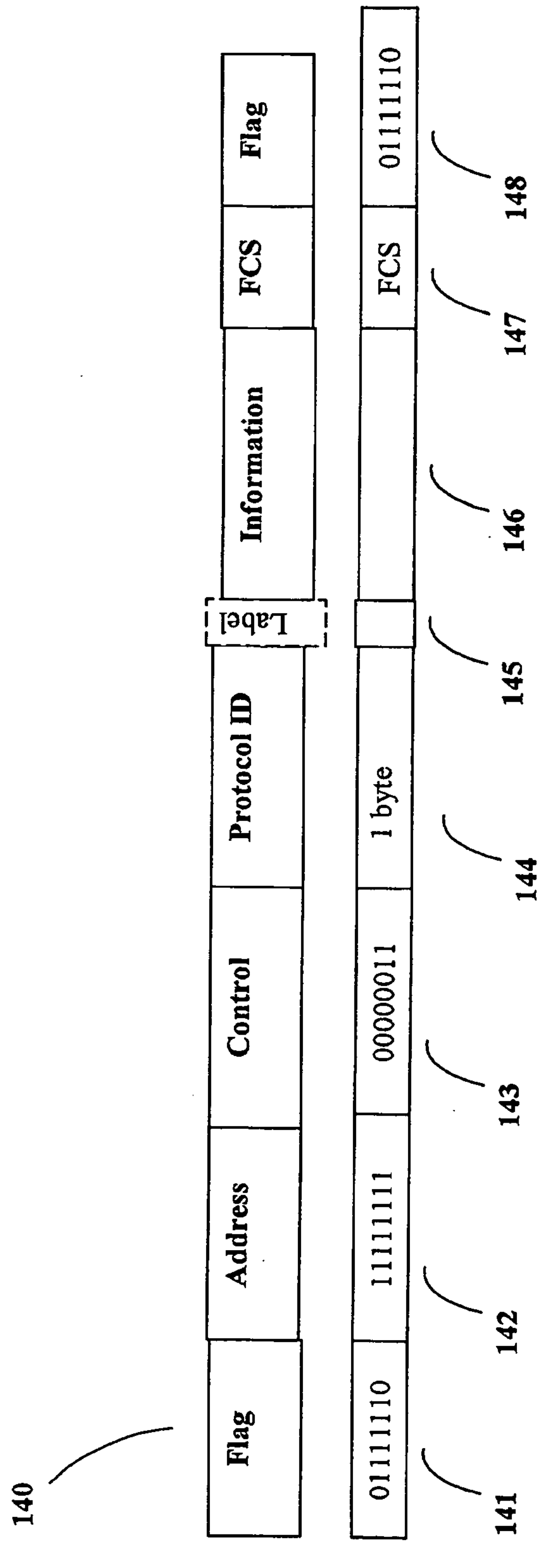


Figure 14. OIR Point-to-Point Packet Format



OIR Point-to-point Frame

Figure 15. InfiniBand Frame Encapsulated within the OIR Point-to-Point Packet

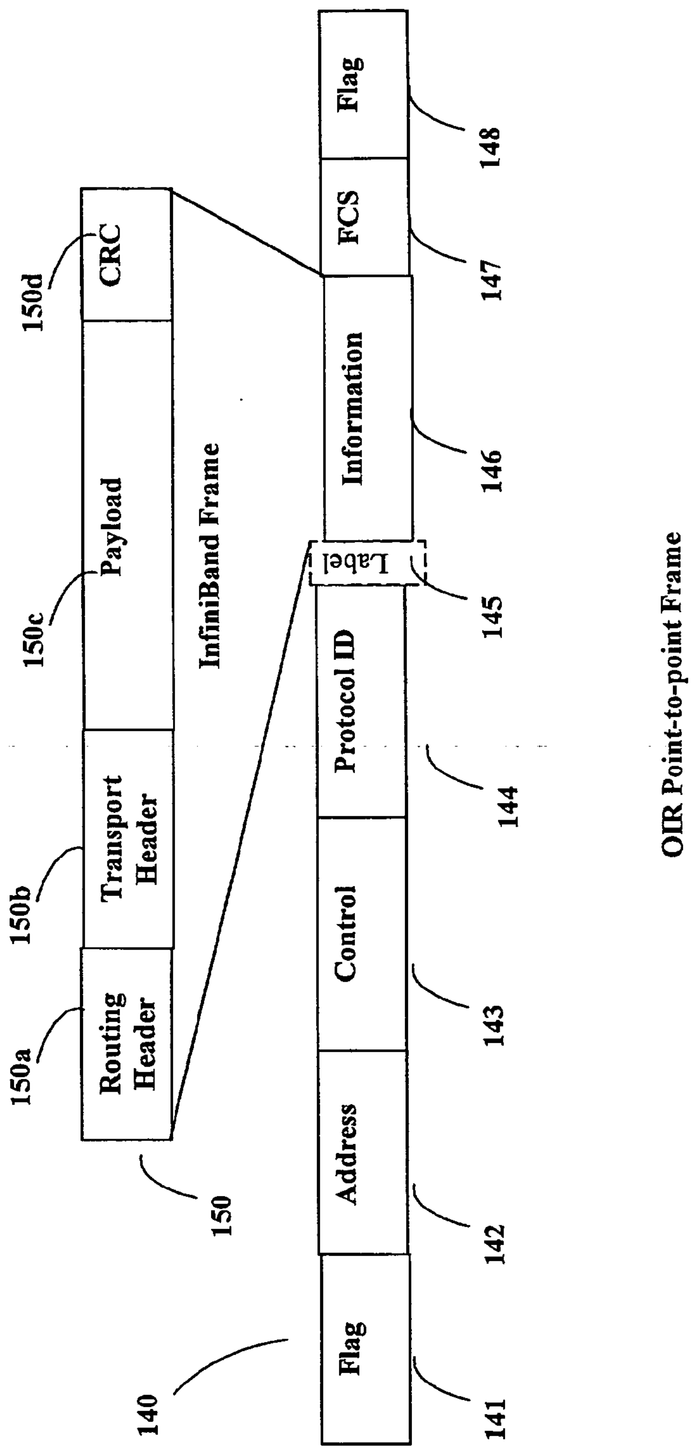


Figure 16. Fibre Channel Frame Encapsulated within the OIR Point-to-Point Packet

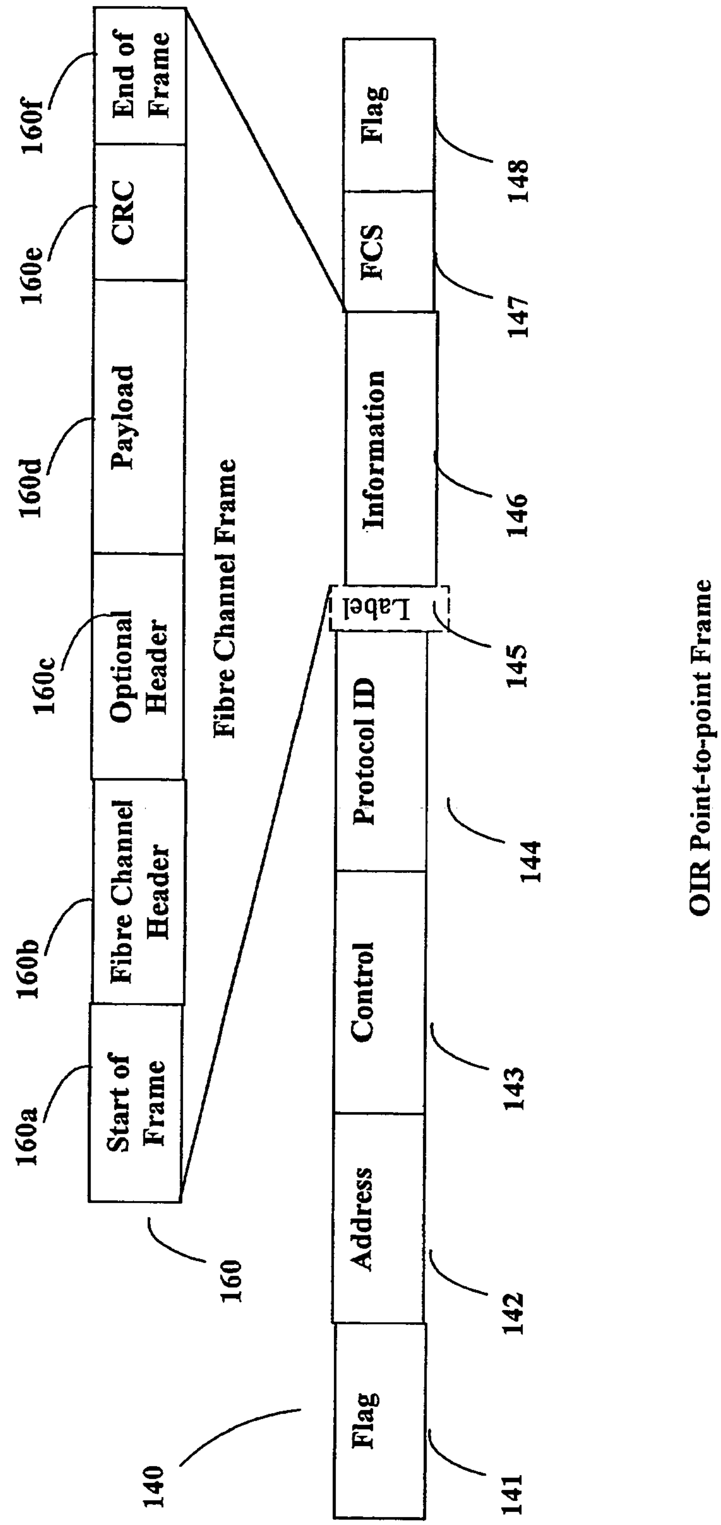


Figure 17. Ethernet Frame Encapsulated within the OIR Point-to-Point Packet

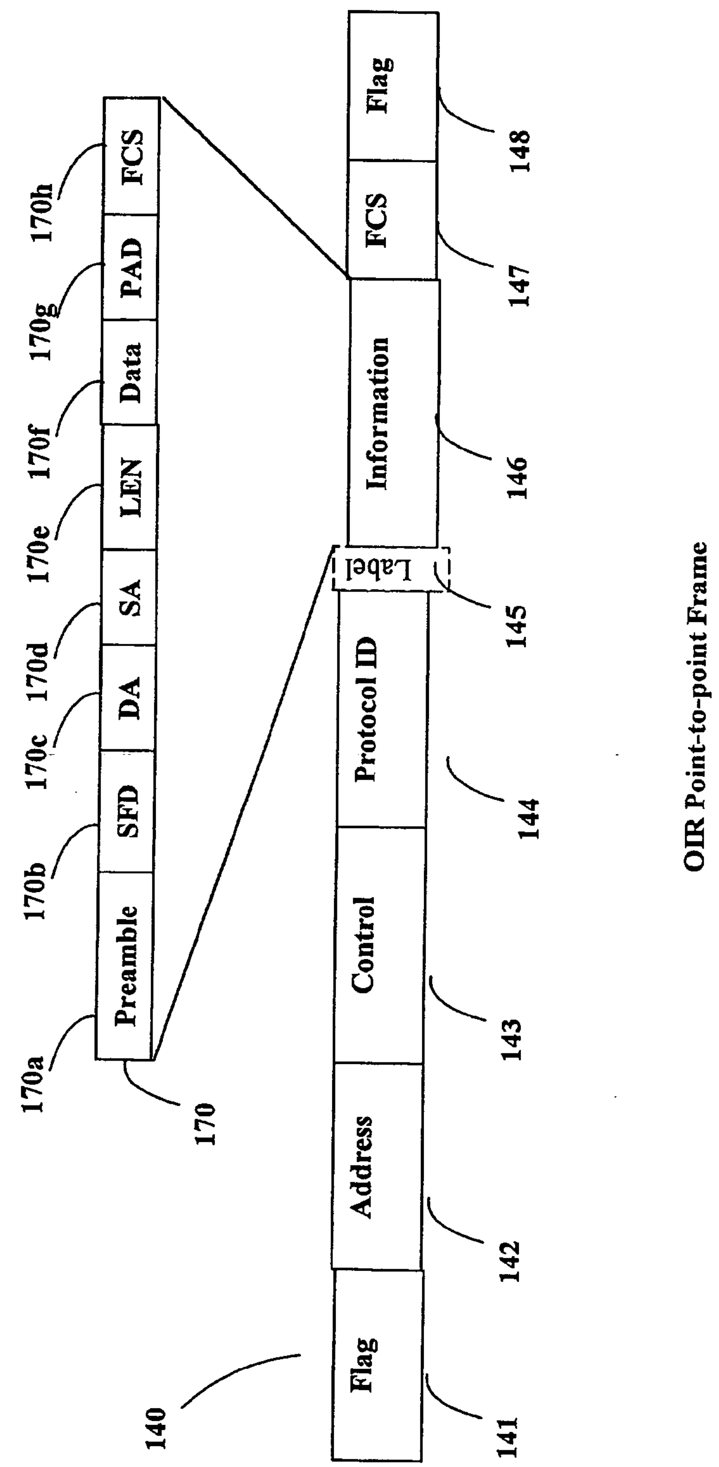
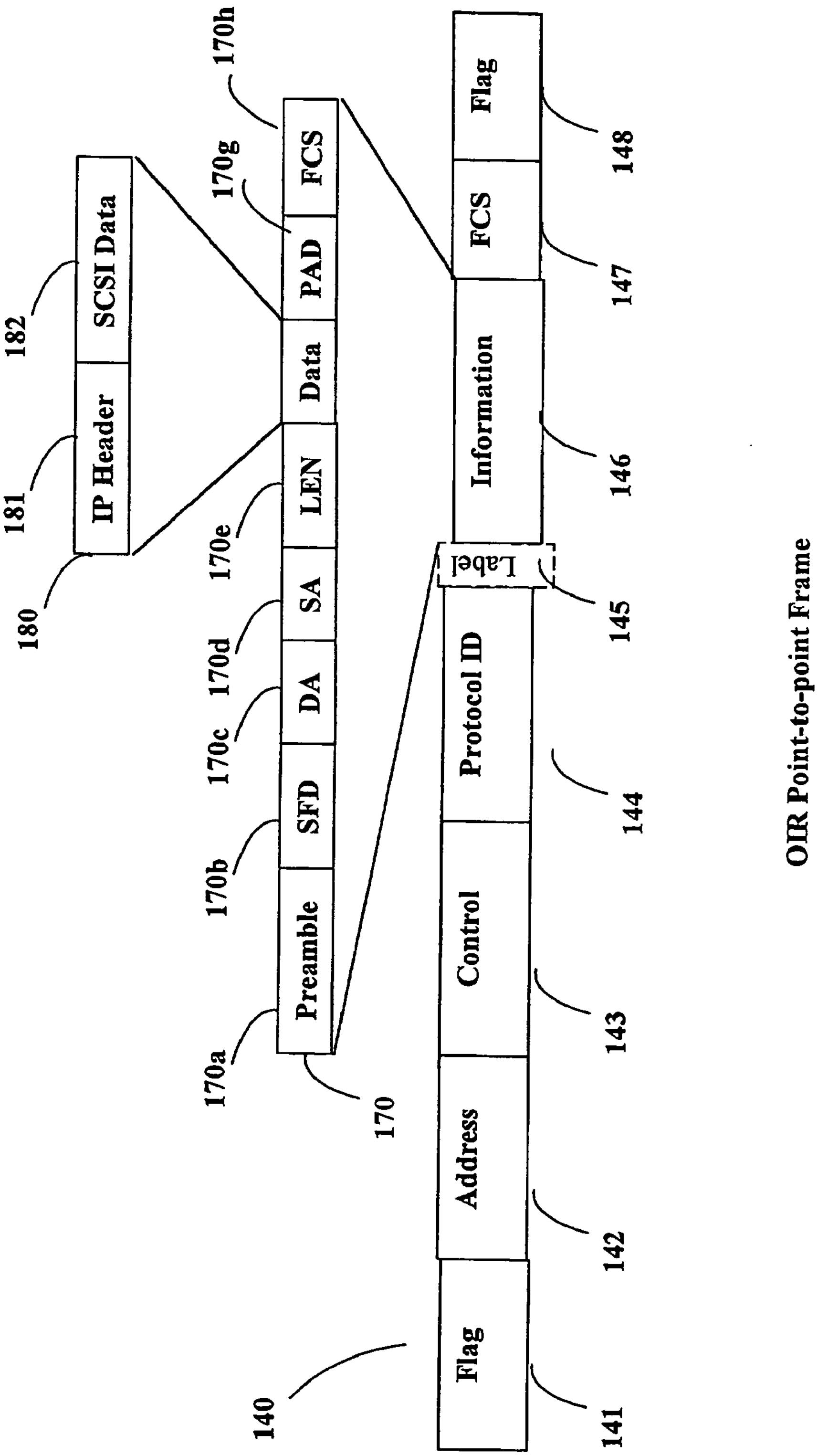


Figure 18. ISCSI Frame Encapsulated within the OIR Point-to-Point Packet



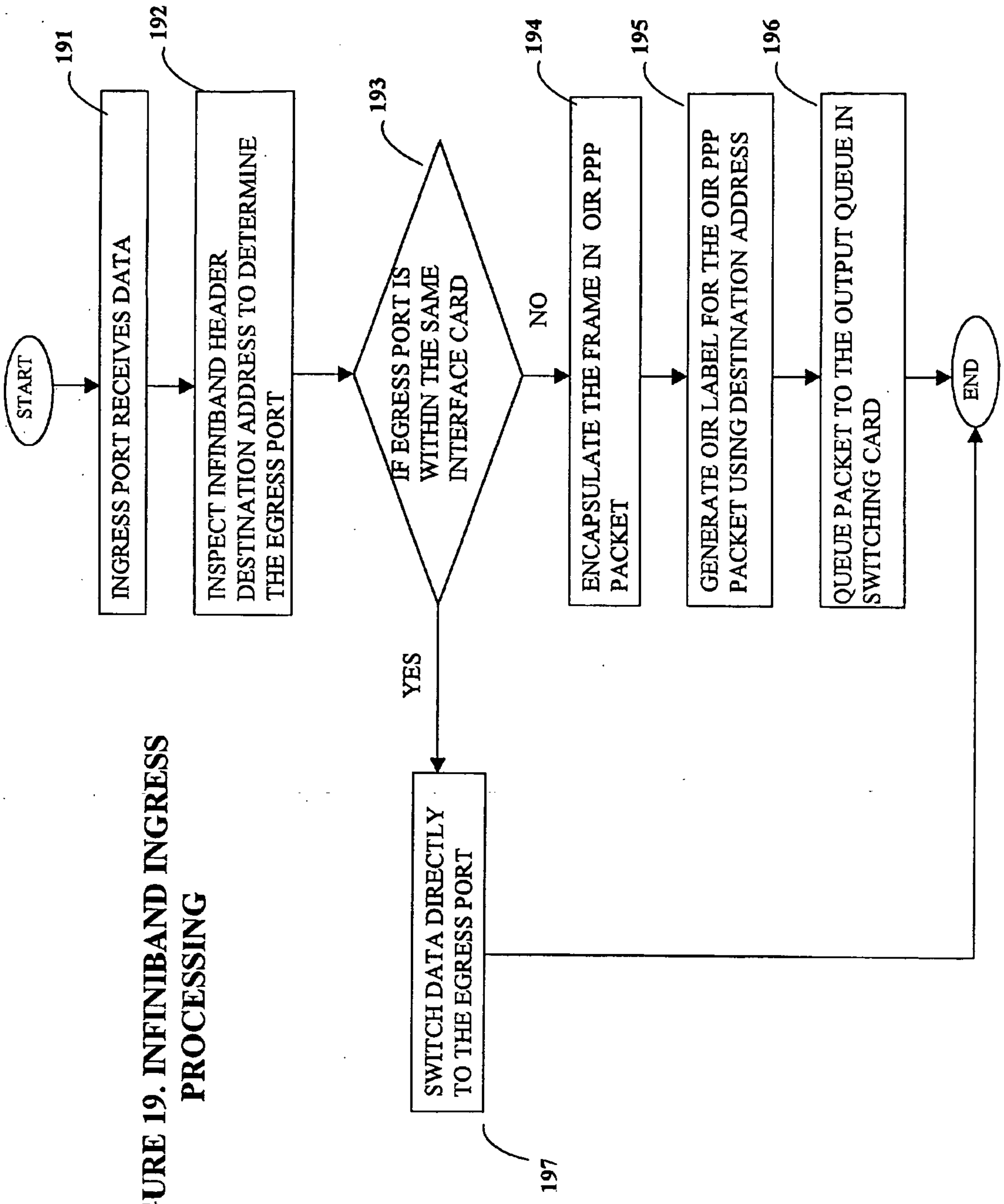
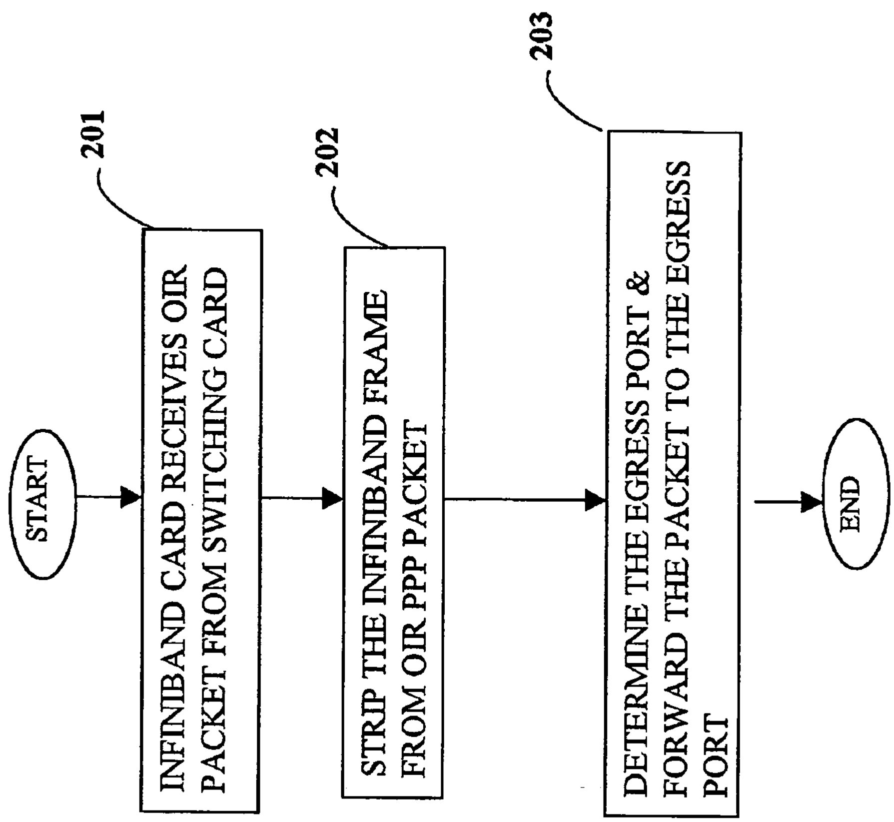


FIGURE 19. INFINIBAND INGRESS PROCESSING

FIGURE 20. INFINIBAND EGRESS PROCESSING



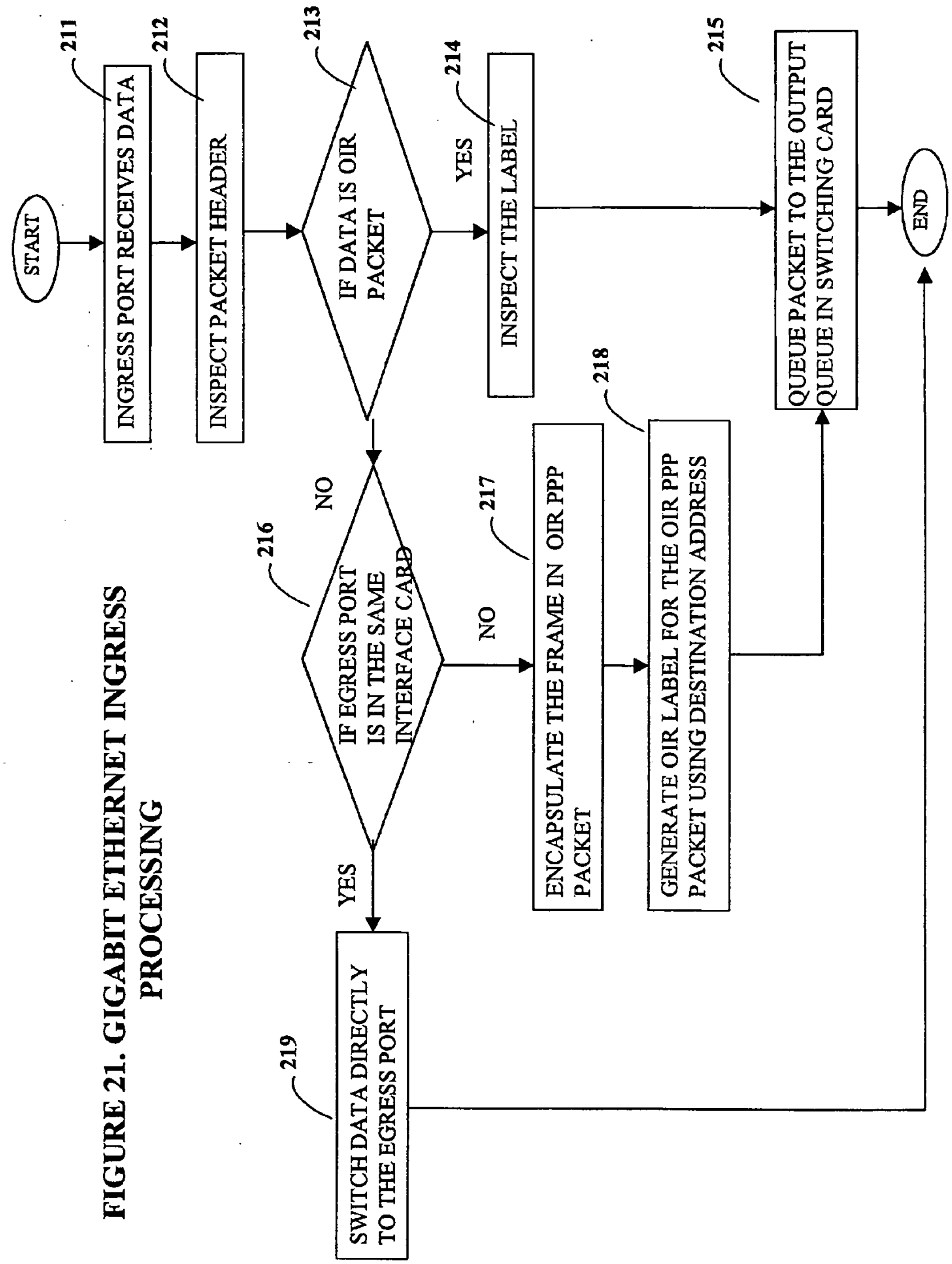


FIGURE 21. GIGABIT ETHERNET INGRESS PROCESSING

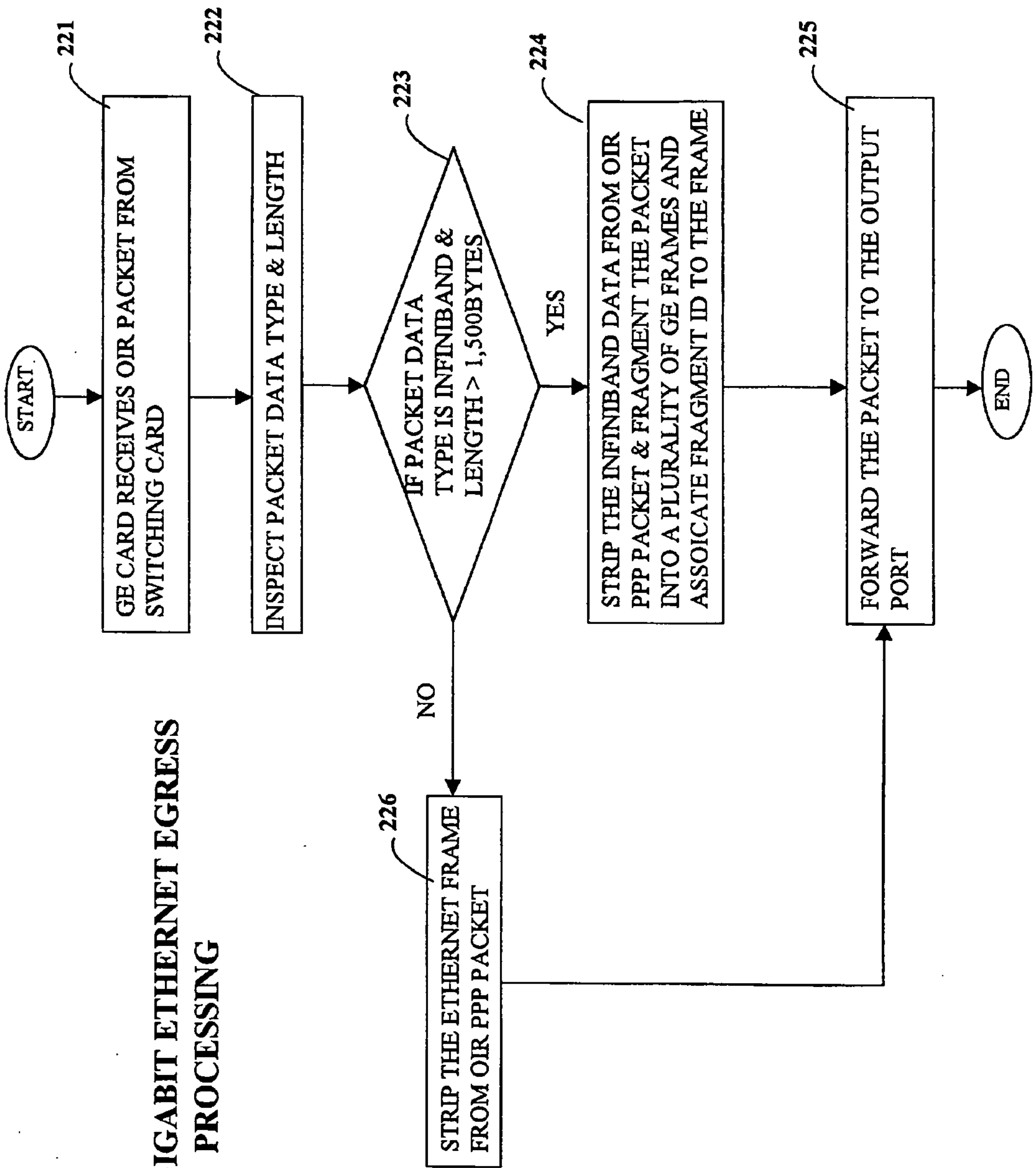


FIGURE 22. GIGABIT ETHERNET EGRESS PROCESSING

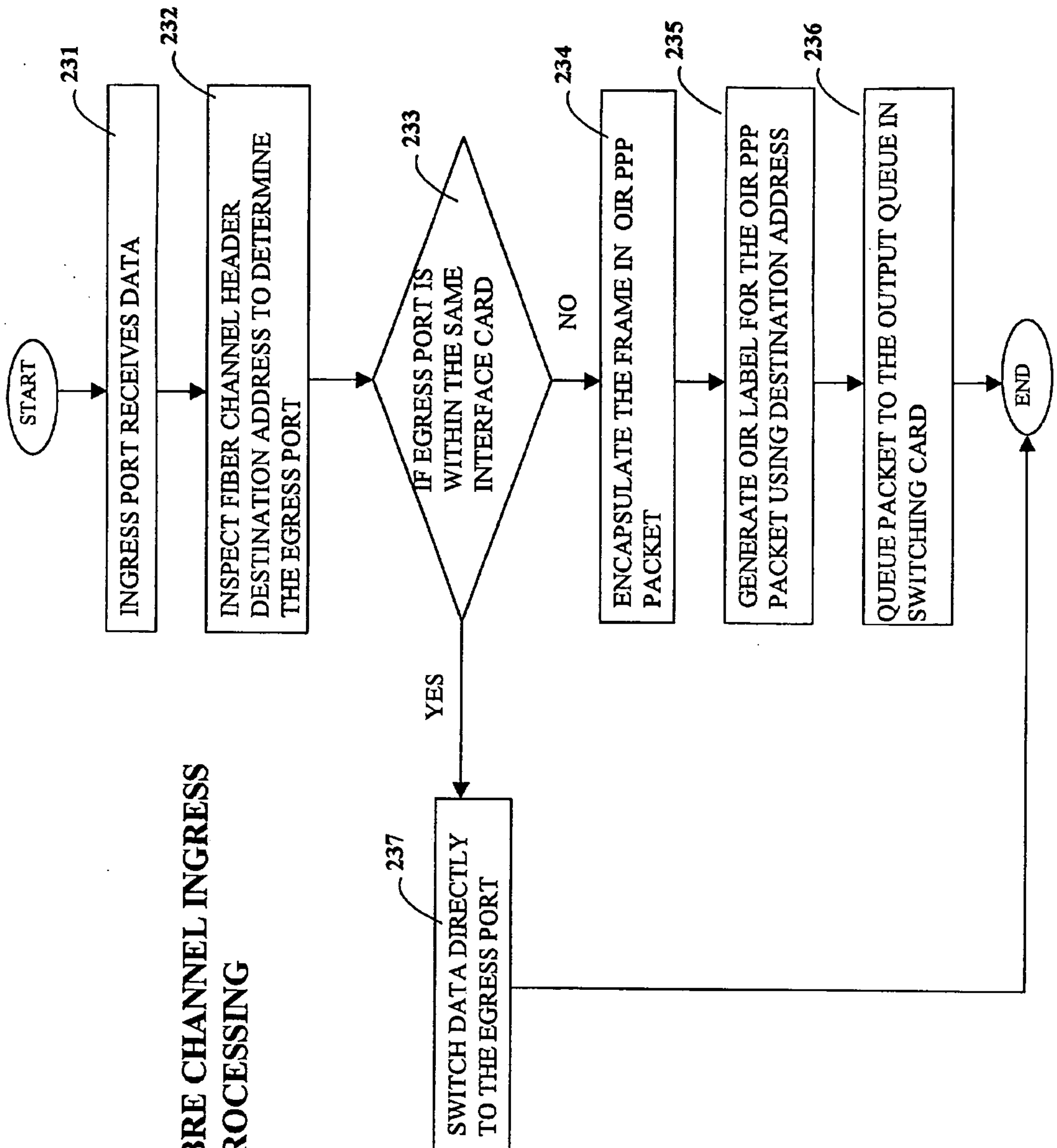
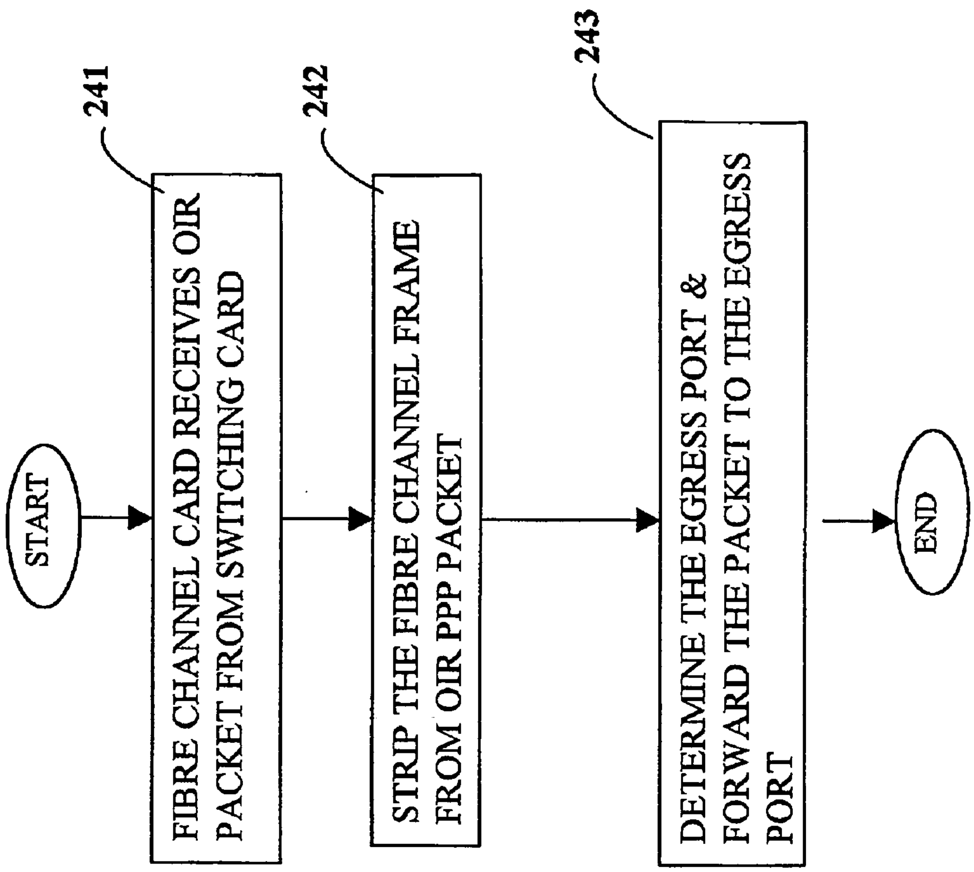


FIGURE 23. FIBRE CHANNEL INGRESS PROCESSING

FIGURE 24. FIBRE CHANNEL EGRESS PROCESSING



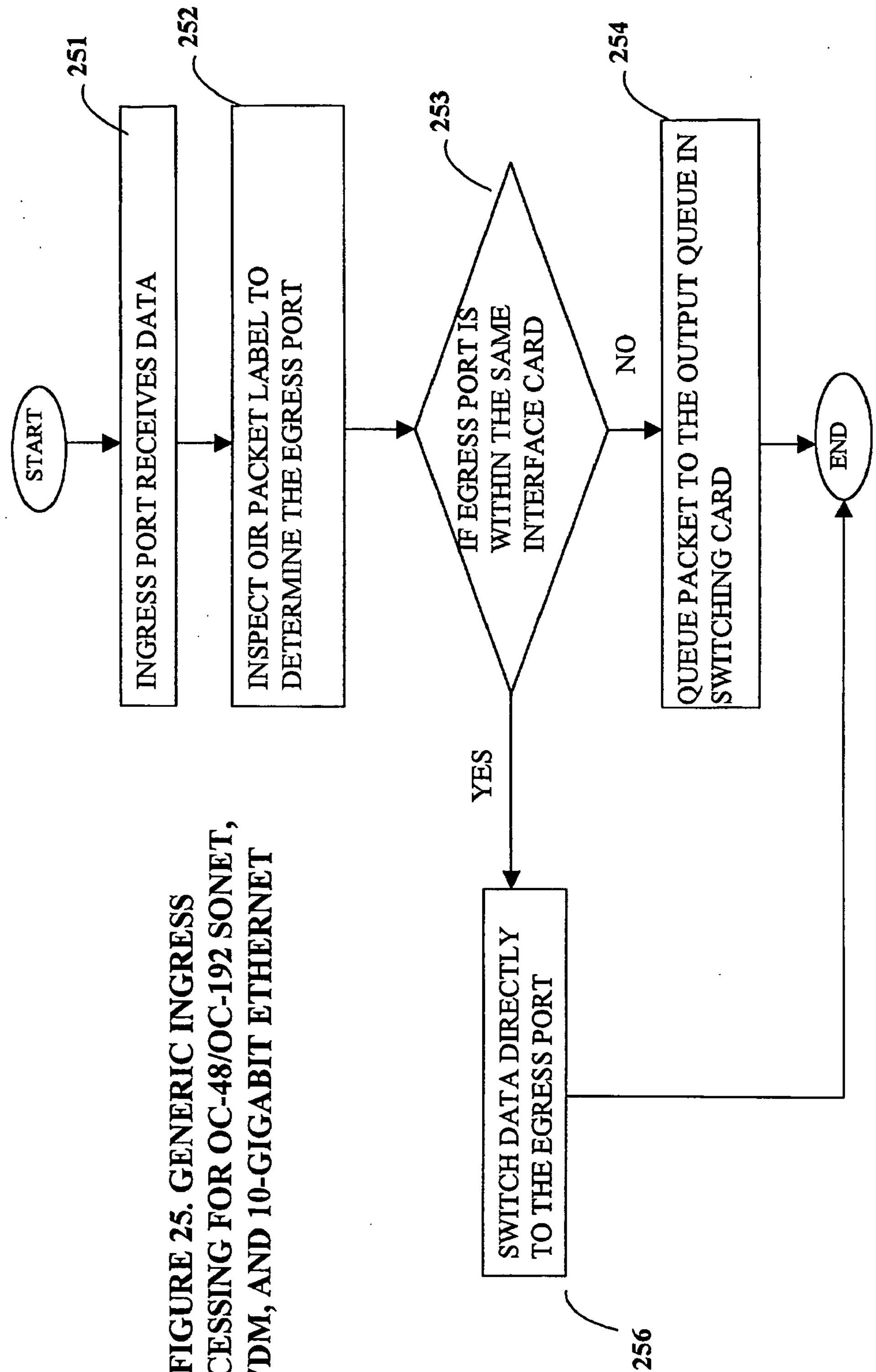
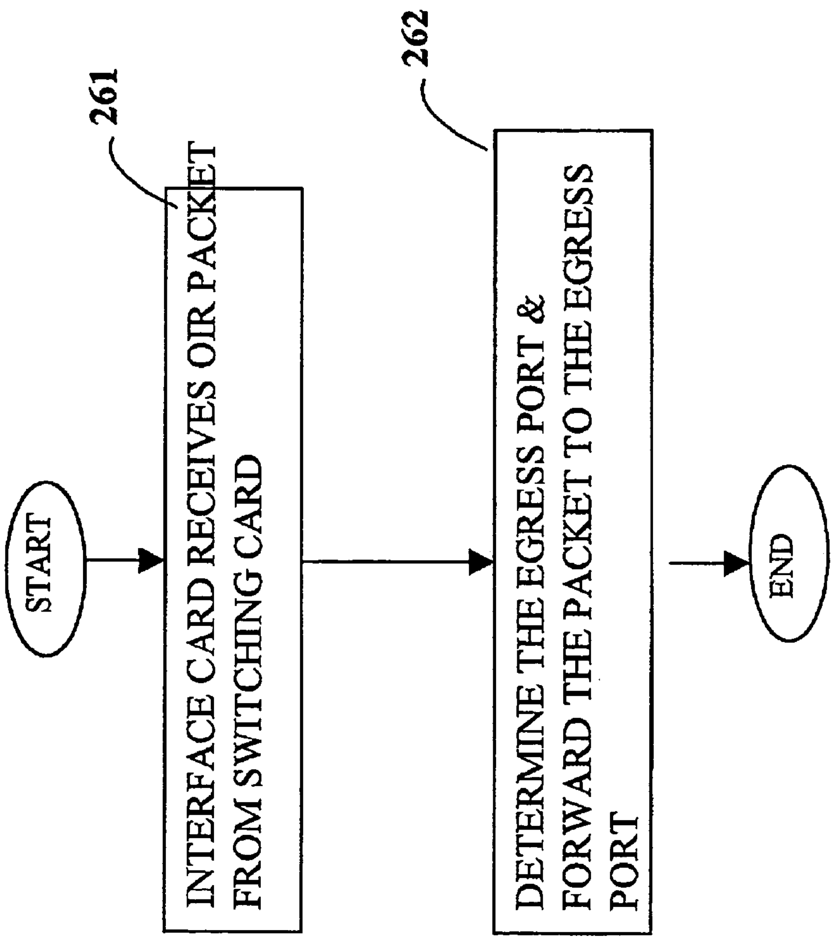


FIGURE 25. GENERIC INGRESS PROCESSING FOR OC-48/OC-192 SONET, DWDM, AND 10-GIGABIT ETHERNET

FIGURE 26. GENERIC INGRESS PROCESSING FOR OC-48/OC-192 SONET, DWDM, AND 10-GIGABIT ETHERNET



MULTI-SERVICE OPTICAL INFINIBAND ROUTER

RELATED APPLICATION

[0001] This application claims the benefit of U.S. Provisional Pat. App. Ser. No. 60/289,274, filed on May 7, 2001. The entire teachings of the above application are incorporated herein by reference.

BACKGROUND

[0002] 1. FIELD OF THE INVENTION

[0003] This invention pertains to a system and method for interconnecting computer devices, networking devices in the local area network, metro area network, wide-area network and system area network using a plurality of computer networking interfaces.

BACKGROUND

[0004] 2. DESCRIPTION OF PRIOR ART

[0005] **FIG. 1** illustrates the Traditional System Architecture. The traditional server contains the processing modules **11**, the I/O modules **12**, and the other interface adapters **13**. The I/O is usually based on the SCSI bus or Fibre Channel. The Host usually "owns" the storage **15**, which is enclosed with the server enclosure **14**. The backup traffic needs to go through the LAN to the server (before getting to another storage device). It has limited scalability (16 devices per bus).

[0006] **FIG. 2** illustrates the InfiniBand System Architecture. When all the major servers joined force to define an Infinite Bandwidth I/O bus, they call it InfiniBand. The idea of the InfiniBand architecture is to decouple the Processing Module, called the Server Host **22**, and the I/O Module, called the target **23**. The Hosts and the Targets are connected through an external switch, called the InfiniBand Switch **22**. This switch can be used to connect to multiple InfiniBand nodes, including IB host, IB target, and other IB switches. The architecture is extremely scalable.

[0007] The InfiniBand is good technology if the user does not have to connect to other nodes outside of the InfiniBand System Area Network. The InfiniBand technology has some limitations; the connection between InfiniBand nodes has to be within 100 meters. In addition, there is no specification for connecting to a network beyond the LAN. For example, there is no interoperability definition for InfiniBand to connect to a SONET network. This is what this invention will be doing. Our goal is to remove these kinds of barriers and evolve InfiniBand to become the complete System Area Network solution to the Application Service Providers, the Storage Service Providers, and the large enterprises.

[0008] **FIG. 3** illustrates the Optical InfiniBand (IB) Architecture when the Optical InfiniBand Router OIR system **31** is used. With this invention, the Optical InfiniBand Router **32**, the IB host **31** can connect to any IB target **34**, **35** without any restrictions. The nodes can be thousands of miles away but the nodes will behave like they are connected through a standard I/O bus. This is the power of our invention and that is why this product is so valuable to target customers.

[0009] In addition to transporting InfiniBand data across Local Area Network (LAN), Metro Area Network (MAN), and Wide Area Network (WAN), it will transport storage system related data across the LAN, MAN and WAN. In prior art, SCSI and Fiber Channel technologies are being used for the Storage Area Network (SAN) transport. This invention will also transport any SAN-based frames, including SCSI and Fibre Channel, across the different networking environment.

[0010] InfiniBand structure and functions are described in the literature and is therefore not described in detail here. Among the relevant reference texts are "InfiniBand Architecture Specification, Release 1.0" (ref. 1) and "InfiniBand Technology Prototypes White Paper" (ref. 15).

[0011] Fibre Channel structure and functions are described in the literature and is therefore not described in detail here. Among the relevant reference texts are "The Fibre Channel Consultant-A Comprehensive Introduction" (ref. 7) and "Fibre Channel-The Basics" (ref. 8).

[0012] Small Computer System Interface (SCSI) structure and functions are described in the literature and is therefore not described in detail here. Among the relevant reference texts are "The Book of SCSI: I/O for the New Millennium" (ref. 17) and "Making SCSI Work" (Ref. 18).

[0013] Gigabit Ethernet structure and functions are described in the literature and is therefore not described in detail here. Among the relevant reference texts are "Media Access Control (MAC) Parameters, Physical Layer, Repeater and Management Parameters for 1000 Mb/s Operation." (Ref. 9), and "Gigabit Ethernet-Migrating to High-Bandwidth LANS" (ref. 8).

[0014] SONET structure and functions are described in the literature and is therefore not described in detail here.

[0015] Among the relevant reference texts are "American National Standard for Telecommunications-Synchronous Optical Network (SONET) Payload Mappings," (ref. 5) and "Network Node Interface for the Synchronous Digital hierarchy (SDH) ," (ref. 6).

[0016] Dense Wavelength Division Multiplexing (DWDM) technology is described in the literature and is therefore not described in detail here. Among the relevant reference texts are "Web ProForum tutorial:DWDM", (ref. 13) and "Fault Detectability in DWDM Systems: Toward Higher Signal Quality & Reliability" (ref. 16).

[0017] Optical technology and Internet Protocol (IP) technologies are described in the literature and are therefore not described in detail here. Among the relevant reference texts are "The Point-to-Point Protocol (PPP)" (ref. 2), "PPP in HDLC-like Framing" (ref. 3), "PPP over SONET/SDH" (ref. 4), "Optical Communication Networks Multi-Protocol Lambda Switching:Combining MPLS Traffic Engineering Control With Optical Cross-Connects, (ref. 11), "Features and Requirements for The Optical Layer Control Plane" (ref. 12).

[0018] In conclusion, insofar as I am aware, no Optical routers or Storage Area System switches formerly developed provides the multi-services interconnection functions with InfiniBand technology. In addition, insofar as I am aware, no networking systems formerly developed provides the gateway function between the InfiniBand devices and the Storage Area Systems devices or Network Attached Storage devices.

SUMMARY OF THE INVENTION

[0019] Objects and Advantages (over the Prior Art)

[0020] Accordingly, besides the objects and advantages of supporting multiple networking/system services described in my above patent, several objects and advantages of the present invention are:

[0021] To provide a system which can extend the transport of InfiniBand from the 100-meters limited to beyond 100 K meters

[0022] To provide a system which can transport InfiniBand data through Gigabit Ethernet interface between the InfiniBand host or target channel devices.

[0023] To provide a system which can transport InfiniBand data through SONET Add-Drop Multiplexer interface between the InfiniBand host or target channel devices.

[0024] To provide a system which can transport InfiniBand data through DWDM interface between the InfiniBand host or target channel devices.

[0025] To provide a system which can provide a gateway function, which can convert InfiniBand data stream to/from Fibre Channel data stream.

[0026] To provide a system which can provide a gateway function, which can transport InfiniBand data stream to/from Network Attached Storage Filer devices.

[0027] To provide a system which can provide Quality of Service control over the InfiniBand data stream through the OIR network. The OIR network can be comprised of Gigabit Ethernet interface, SONET interfaces, Fibre Channel interfaces and DWDM interfaces.

[0028] Further objects and advantages are to provide a highly reliable, highly available, and highly scalable system, which can be upgradeable to different transport services, including Gigabit Ethernet, SONET, and DWDM. The system is simple to use and inexpensive to manufacture compare to the current Gigabit Ethernet based IP routers, SONET Add-Drop Multiplexers, and DWDM devices. Still further objects and advantages will become apparent from a consideration of the ensuing description and drawings.

Objects (Benefits) to our Customers

[0029] This invention provides our customers with the needed performance and the benefits as follows:

[0030] Simplification

[0031] This invention combines the capability of the InfiniBand, Gigabit Ethernet, SONET, and DWDM into one power router. By providing the multi-services, the customers can easily upgrade and modify the system/network infrastructure without major installation delay or training requirements.

[0032] Providers can greatly simplify service delivery by bringing InfiniBand, Gigabit Ethernet, SONET, DWDM service directly to every midsize to large enterprise and major application service provider (ASP)/Web hosting center.

[0033] Reliability

[0034] The OIR provides redundant hardware platform and traffic paths. By using SONET Automatic Protection Systems or DWDM optical redundant path protection methods, the OIR network is guaranteed to recover from any line/path or hardware failure within 50 milliseconds. The fast failure recovery capability is the key advantage that OIR has over the existing Ethernet based networks.

[0035] Quality of Service (QoS) support

[0036] The customers can configure the user traffic based on their needs. Policy-based Network Management provided with the OIR can manage traffic to each user connection (micro-flows). The OIR supports policies to define deterministic, guaranteed, assured, and shared traffic.

[0037] Scalable Performance

[0038] The OIR can be scaled up using interchangeable line cards. To complement the existing infrastructure, the LAN/SAN/NAS services can be connected to the OIR. Multi-service traffic can be aggregated into high speed Gigabit Ethernet (3 Gbps to 10 Gbps), SONET (2.5 Gbps to 10 Gbps), or multiple wavelength DWDM (up a multitude of gigabits per second) systems.

BRIEF DESCRIPTION OF THE DRAWINGS

[0039] **FIG. 1**. is a block diagram illustrating a traditional server system architecture.

[0040] **FIG. 2**. is a block diagram illustrating the InfiniBand Architecture.

[0041] **FIG. 3**. is a block diagram illustrating the Optical InfiniBand Routing (OIR) system.

[0042] **FIG. 4**. is a block diagram illustrating an OIR sample system layout.

[0043] **FIG. 5**. is a block diagram illustrating the OIR Logical Multi-Services System Layout.

[0044] **FIG. 6**. is a block diagram illustrating a method for inter-networking System Area Network (SAN) switching using OIR technology.

[0045] **FIG. 7**. is a block diagram illustrating a method for InfiniBand Packet switching through the OIR system.

[0046] **FIG. 8**. is a block diagram illustrating a method for Inter-OIR InfiniBand Packet switching using Gigabit Ethernet Interfaces.

[0047] **FIG. 9**. is a block diagram illustrating a method for Inter-OIR InfiniBand Packet switching using SONET Interfaces.

[0048] **FIG. 10**. is a block diagram illustrating a method for Inter-OIR InfiniBand Packet switching using DWDM Interfaces.

[0049] **FIG. 11**. is a block diagram illustrating a method for Inter-OIR Fibre Channel Data switching using DWDM Interfaces.

[0050] **FIG. 12**. is a block diagram illustrating a method for Inter-OIR InfiniBand/Fibre Channel Data switching using DWDM Interfaces.

[0051] **FIG. 13**. is a block diagram illustrating a method for Inter-OIR InfiniBand/iSCSI Data switching using DWDM Interfaces.

[0052] **FIG. 14**. is a block diagram illustrating Packet Format for the OIR system.

[0053] **FIG. 15**. is a block diagram illustrating the InfiniBand Frame encapsulated within the OIR Packet.

[0054] **FIG. 16**. is a block diagram illustrating the Fibre Channel Frame encapsulated within the OIR Packet.

[0055] **FIG. 17**. is a block diagram illustrating the Ethernet Frame encapsulated within the OIR Packet.

[0056] **FIG. 18**. is a block diagram illustrating the iSCSI Frame encapsulated within the OIR Packet.

[0057] **FIG. 19**. is a block diagram illustrating the InfiniBand Ingress Processing

[0058] FIG **20**. is a block diagram illustrating the InfiniBand Egress Processing

[0059] FIG **21**. is a block diagram illustrating the Gigabit Ethernet Ingress Processing

[0060] FIG **22**. is a block diagram illustrating the Gigabit Ethernet Egress Processing

[0061] FIG **23**. is a block diagram illustrating the Fibre Channel Ingress Processing

[0062] FIG **24**. is a block diagram illustrating the Fibre Channel Egress Processing

[0063] FIG **25**. is a block diagram illustrating the Generic Ingress Processing for OC-48 SONET interface, OC-192 SONET interface, DWDM interface, and 10-Gigabit Ethernet interface.

[0064] FIG **26**. is a block diagram illustrating the Generic Egress Processing for OC-48 SONET interface, OC-192 SONET interface.

Reference Numerals In Drawings

[0065] **11** Processing Module

[0066] **12** PCI Bus Interface

[0067] **13** Input/Output Controller

[0068] **14** Traditional Server (Enclosure)

[0069] **15** MultiMedia Device

[0070] **16** Local Area Network

[0071] **17** Storage (Disks, Tapes, Flash Memory)

[0072] **18** Graphics Device

[0073] **21** InfiniBand Server Host

[0074] **22** InfiniBand Switch

[0075] **23** InfiniBand Target Channel Adapter

[0076] **31** Optical InfiniBand Router (OIR System)

[0077] **31*a*** Originating OIR System (same as 31-OIR system with infiniBand interface support)

[0078] **31*b*** Intermediate OIR System (same as 31-OIR system with Gigabit Ethernet interface support)

[0079] **31*c*** Originating OIR System (same as 31-OIR system with SONET interface support)

[0080] **31*d*** Destined OIR System (same as 31-OIR system with DWDM interface support)

[0081] **32** 2 Fiber/4 Fiber SONET/DWDM Ring Network

[0082] **41** Management Card (Active/Standby)

[0083] **42** InfiniBand Interface Card

[0084] **43** DWDM Interface Card

[0085] **44** OC-48 SONET Card

[0086] **45** OC-192 SONET Card

[0087] **46** 10-Gigabit Ethernet Card

[0088] **47** Ether-Channel Interface Card (1-Gigabit Ethernet Interface Card)

[0089] **48** Fiber Channel Interface Card

[0090] **49** Switching Fabric Card (Active/Standby)

[0091] **51** Gigabit Ether-Channel Processing System

[0092] **52** 10-Gigabit Ethernet Processing System

[0093] **53** OC-48 SONET Processing System

[0094] **54** DWDM Processing System

[0095] **55** InfiniBand Processing System

[0096] **56** Fibre Channel Processing System

[0097] **57** OC-192 SONET Processing System

[0098] **58** Management Processing System

[0099] **59** Switching Processing System

[0100] **61*a*** Client Applications/ Upper Level Protocols

[0101] **61*b*** InfiniBand Operations/ Transport Layer

[0102] **61*c*** Network Layer

[0103] **61*d*** Link Encoding within Link Layer

[0104] **61*e*** Media Access Control within Link Layer

[0105] **61*f*** Optics Fiber(O)/ Physical Layer

[0106] **62*a*** InfiniBand Device/End Node

[0107] **62*b*** FibreChannel Device/End Node

[0108] **62*c*** iSCSI Device/End Node

[0109] **63** InfiniBand Interface on OIR System

[0110] **64** Gigabit Ether-Channel Interface on OIR System

[0111] **65** SONET Interface on OIR System

[0112] **66** 10-Gigabit Ethernet Interface on OIR System

[0113] **67** DWDM Interface on OIR System

[0114] **68** Fibre Channel Interface OIR System

[0115] 69 Switching Processing System on OIR System (performing packet relay)

[0116] 111*a* Generic Client Applications/ Upper Level Protocols

[0117] 111*b* Fibre Channel Link Encapsulation

[0118] 111*c* Fibre Channel Common Services

[0119] 111*d* Fibre Channel Exchange and Sequence Management

[0120] 111*e* Fibre Channel 8*b*/10*b* Encode/Decode and Link Control

[0121] 111*f* Fibre Channel Optics Fiber(O)/ Physical Layer

[0122] 121 InfiniBand/Fibre Channel Gateway

[0123] 131 InfiniBand/iSCSI Gateway

[0124] 132*a* iSCSI Operation

[0125] 132*b* Ethernet Link Encoding

[0126] 132*c* Ethernet Media Access Control

[0127] 132*d* Ethernet Optics Fiber(O)/ Physical Layer

[0128] 140 OIR System Point-to-Point Format

[0129] 141 Frame Start Flag Field within OIR Point-to-Point Frame

[0130] 142 Address Field within OIR Point-to-Point Frame

[0131] 143 Control Field within OIR Point-to-Point Frame

[0132] 144 Protocol Identifier Field within OIR Point-to-Point Fame

[0133] 145 Label Field within OIR Point-to-Point Frame

[0134] 146 Information Field within OIR Point-to-Point Frame (Data Payload)

[0135] 147 Frame Check Sequence Field within OIR Point-to-Point Frame

[0136] 148 Frame End Flag Field within OIR Point-to-Point Frame

[0137] 150 InfiniBand Frame Format

[0138] 150*a* Routing Header Field within InfiniBand Frame

[0139] 150*b* Transport Header Field within InfiniBand Frame

[0140] 150*c* Payload Field within InfiniBand Frame

[0141] 150*d* CRC Field within InfiniBand Frame

[0142] 160 Fibre Channel Frame

[0143] 160*a* Start of Frame Field within Fibre Channel Frame

[0144] 160*b* Fibre Channel Header Field within Fibre Channel Frame

[0145] 160*c* Optional Header Field within Fibre Channel Frame

[0146] 160*d* Payload Field within Fibre Channel Frame

[0147] 160*e* CRC Field within Fibre Channel Frame

[0148] 160*f* Start of Frame Field within Fibre Channel Frame

[0149] 170 Ethernet Frame

[0150] 170*a* Preamble Field within Ethernet Frame

[0151] 170*b* Start Frame Delimiter (SFD) Field within Ethernet Frame

[0152] 170*c* Destination Address (DA) Field within Ethernet Frame

[0153] 170*d* Source Address (SA) Field within Ethernet Frame

[0154] 170*e* Length (LEN) Field within Ethernet Frame

[0155] 170*f* Data Field within Ethernet Frame

[0156] 170*g* Padding Field within Ethernet Frame

[0157] 170*h* Frame Check Sequence Field within Ethernet Frame

[0158] 180 Internet Protocol Packet Format

[0159] 181 Internet Protocol Header

[0160] 182 SCSI Data

[0161] 191-262 Labels for the Data Flow Diagrams

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0162] The invention, an InfiniBand Optical Router, has the capabilities to transport and route data packets to and from the following devices:

[0163] InfiniBand Host Server device

[0164] InfiniBand Target Channel device

[0165] SONET Add-Drop Multiplexing device

[0166] DWDM device

[0167] Gigabit Ethernet-based IP Switching device

[0168] Gigabit Ethernet-based IP Routing device

[0169] Fiber Channel Host Channel Adapter device

[0170] ISCSI device

DRAWINGS

FIGS. 4 and 5—PREFERRED EMBODIMENT

[0171] **FIG. 4** illustrates a sample physical system layout and **FIG. 5** illustrates the logical system layout of the Optical InfiniBand Routing (OIR) device **31**. Each type of line card will contain different layer **1** and layer **2** hardware components. For example, the OC-48 SONET cards **44** will have an optical transceiver and SONET framer while the

Ethernet cards **47** will have Ethernet transceivers with MAC/GMAC interface. The OIR device contains the following:

[0172] Management Card(s) **41**—are responsible for the management and control of the OIR system. In addition to the OIR management functions, the Management Processing System **58** can be enhanced to perform higher-level application functions as needed.

[0173] InfiniBand Interface Card(s) **42**—are responsible for interfacing with the InfiniBand Host and Target Channel devices. The InfiniBand Processing System **55** processes the InfiniBand data and encapsulates the InfiniBand payload into the OIR Point-to-Point Packet format **140**.

[0174] DWDM Interface Card(s) **43**—are responsible for interfacing with upstream or downstream DWDM system. The function of the DWDM Processing system **54** is mainly for multiplexing and de-multiplexing lower speed data packets onto the high-speed DWDM optical transport.

[0175] OC-48 SONET Card(s) **44**—are responsible for interfacing with upstream or downstream OC-48 SONET system. The function of the SONET Processing system **53** is mainly for transporting SONET payload between SONET capable devices, including OIR system **31**. Traffic from the SONET card **44** is de-multiplexed, de-framed and packet extracted before sending to the Network Processor for packet processing. The SONET Processing System **53** will perform path, line, and section overhead processing and pointer alignment processing.

[0176] OC-192 SONET Card(s) **45**—are responsible for interfacing with upstream or downstream OC-192 SONET system. The function of the SONET Processing system **57** is mainly for transporting SONET payload between SONET capable devices, including OIR system **31**, and multiplexing and de-multiplexing lower speed data packet onto the high-speed OC-192 SONET optical transport.

[0177] Gigabit Ether-Channel Card(s) **47**—are responsible for interfacing with upstream or downstream Gigabit Ethernet system or OIR Gigabit Ether-Channel Interfaces **47**. The Gigabit Ethernet card will support the GBIC interface to allow for serial data transmission over fiber optic or coaxial cable interfaces. The Gigabit Ether-Channel Processing System **51** processes the Ethernet data and encapsulates the Ethernet payload into the OIR Point-to-Point Packet format **140**. It also performs fragmentation and de-fragmentation function on InfiniBand frame or other payload that has large frame size than Ethernet frame. The fragmented frames are forwarded to the destination within the OIR system **31** by a plurality of Gigabit Ethernet frames. The fragmented frames are reassembled (or de-fragmented) at the destination Gigabit Ether-Channel Interface **47** of the OIR system **31**.

[0178] When InfiniBand traffic is transported through the OIR system **31** to another OIR system **31** within the OIR network, the Gigabit Ether-Channel Processing system **51** will activate the Ether-Channel processing function to transport the InfiniBand data packet using a plurality of Gigabit Ethernet channels. The Gigabit Ethernet Processing system **51** is responsible for fragmenting the InfiniBand data frame into smaller Ethernet packets and de-fragmenting the Ethernet packets into the original InfiniBand data frame.

[0179] When Fibre Channel traffic is transported through OIR system **31** to another OIR system **31** within the OIR network, the Gigabit Ether-Channel Processing system **51** will activate the Ether-Channel processing function to transport the Fibre Channel data packet using a plurality of Gigabit Ethernet channels. The Gigabit Ethernet Processing system **51** is responsible for fragmenting the Fibre Channel data frame into smaller Ethernet packets and de-fragmenting the Ethernet packets into the original Fibre Channel data frame.

[0180] When IP traffic is transported through the OIR network, no special Ether-Channel function will be used. The IP traffic will be packeted into the OIR packet format to be transported between OIR systems **31**.

[0181] When iSCSI traffic is transported through the OIR network, no special Ether-Channel function will be used. The iSCSI traffic will be encapsulated within the IP payload, and then the IP payload will be packeted into the OIR packet format to be transported between OIR systems **31**.

[0182] 10-Gigabit Ethernet Interface Card(s) **46**—are responsible for interfacing with upstream or downstream 10-Gigabit Ethernet systems. The function of the 10-Gigabit Ethernet Processing System **52** is mainly for transporting 10-Gigabit Ethernet Frames between 10-Gigabit Ethernet capable devices, including OIR system **31**, and multiplexing and de-multiplexing lower speed data packets onto the high-speed 10-Gigabit Ethernet optical transport.

[0183] Fibre Channel Interface Card(s) **48**—are responsible for interfacing with the Fibre Channel capable Channel devices. The Fibre Channel Processing System **56** processes the Fibre Channel data and encapsulates the Fibre Channel frames into the OIR Point-to-Point Packet Format **140**.

[0184] Switching Fabric Cards(s) **49**—are responsible for performing arbitration amongst packets from different input sources. Based on the Quality of Service policies, the Switching Processing System **59** will schedule the packets to be transported to different output ports of different interface cards.

OPERATIONS—FIGS. **6**, **7**,**8**,**9**,**10**,**11**,**12**,**13**

[0185] **FIG. 6** is a block diagram illustrating how InfiniBand (IB) data can be transported through the OIR system **31** to other InfiniBand devices. As is known in the prior art, the Open System Interconnection ("OSI") model is used to describe computer network. The OSI model consists of seven layers: physical, link, network, transport, session, presentation, and application. Since the OIR is a routing device that focuses on the network and link layer, the other 5 layers will not be discussed in detail.

[0186] In a normal InfiniBand operation, the client application **61*a*** at the originating end nodes **62*a*** invokes an IB operation **61*b*** on an InfiniBand capable device, an Infini-

Band Host Channel Adapter. The Host Channel Adapter interprets the Work Queue Elements (WQE), creates a request packet with the appropriate destination address. The destination address is composed of two unicast identifiers—a Global Identifier (GID) and Local Identifier (LID). The GID is used by the network layer **61***c* for routing the packets between subnets. The LID is used by the Link Layer **61***d* to switch packets within a subnet.

[0187] The physical layer **61***f* is responsible for establishing physical link and delivering received control and data bytes to the link layer **61***d*, **61***e*. The Link Layer **61***d*, **61***e* provides supports for addressing, buffering, flow control, error detection and switching. The InfiniBand request packet is sent from the originating end node **31***a* to the OIR InfiniBand Interface Card **42** of an OIR system **31***b*.

[0188] The OIR InfiniBand Processing System **55** encapsulates the InfiniBand packet into the OIR Packet payload **150***c*. In addition, it will generate an OIR label **145**, which is used by the OIR system **31** to route the InfiniBand packet to the destination end node **31***b*.

[0189] In **FIG. 6**, the originating OIR node **31***a* and intermediate OIR node **31***b* are interfacing using Gigabit Ethernet interfaces **64**. Therefore, the Gigabit Ether-Channel Processing System **51** within the OIR nodes **31***a* will convert the Inf iniBand packet into a plurality of smaller Ethernet frames before encapsulating it into the OIR payload. The receiving OIR node **31***b* will reassemble the Ethernet frames into a complete InfiniBand packet.

[0190] **FIG. 6** demonstrates that when the intermediate OIR nodes **31***b* and **31***c* are using SONET interfaces **65**, the InfiniBand packet will be encapsulated within an OIR payload and transported using the SONET interface **65**.

[0191] Another sample transport demonstrated in **FIG. 6** is the 10-Gigabit Ethernet interface **66** between the intermediate OIR nodes **31***c* and the destined OIR node **31***d*. The OIR payload, which contains the InfiniBand packet encapsulated within, will be transported directly on the 10-Gigabit Ethernet interface **66** to OIR node **31***c* without further processing. At the destined OIR node **31***d*, the InfiniBand packet will be forwarded to the destined port on the InfiniBand Interface card **42** to be transported to the InfiniBand end node **62***a*.

[0192] **FIG. 7** illustrates the method of how the InfiniBand packets are switched using the OIR system **31**.

[0193] From the InfiniBand client's **61***a* point of view, the InfiniBand Host Operations **61***b* can be performed directly on the InfiniBand Target **62***a*. The details of how the InfiniBand Work Requests are performed are transparent to the Client **61***a*. The actual operation in packet relaying is done by the OIR system **31**.

[0194] From an operational point of view, the InfiniBand end nodes **62***a* are connected to a true InfiniBand switch as defined in the InfiniBand Architecture Specification (see reference [1]), although the OIR system **31** provides a multitude of InfiniBand ports than any existing InfiniBand switching device. The InfiniBand card **42** will detect whether the connecting InfiniBand end nodes is an InfiniBand host (through its Host Channel Adapter interface) or an InfiniBand target (through its Target Channel Adapter interface) and set up the link accordingly. The Packet relay function **69** is provided by the OIR system **31** to switch InfiniBand packets from one InfiniBand interface port **63** to another interface port **63** within the same interface card **42** or to another interface card on the same OIR system **31**.

[0195] **FIG. 8** illustrates the method of how the InfiniBand packets are transported through the OIR nodes **31***a*, **31***b* using the Gigabit Ether-Channel interfaces **65**. The Gigabit Ether-Channel is composed of a plurality of 1-Gigabit Ethernet interfaces **65**. The multiple 1-Gigabit Ethernet bandwidth is aggregated into a logical channel to support the higher bandwidth that is received from the InfiniBand interface. The fragmentation and de-fragmentation functions are performed by the Gigabit Ether-Channel processing system **51**.

[0196] The InfiniBand end nodes **62***a* can interface to the OIR system **31***a*, **31***b* using a single InfiniBand fiber link. The OIR system **31***a*, **31***b* will in turn fragment and de-fragment the InfiniBand frames into multiple 1-Gigabit Ethernet frame before passing them between the OIR systems **31***a*, **31***b*. The assignment of the 1-Gigabit Ethernet ports to the Ether-Channel can be provisioned by the user or can be done using the default configuration.

[0197] **FIG. 9** illustrates the method on how the InfiniBand packets are routed through the OIR system **93**,**94** using the SONET interface. InfiniBand frames transported over SONET use the Point-to-Point protocol, based on IETF Packet over SONET (see reference [2], [3], and [4]). PPP protocol uses the SONET transport as a byte-oriented full-duplex synchronous link. The OIR Point-to-Point Packet **140** is mapped into the SONET Synchronous Payload Envelope (SPE) based on the payload mapping. The packet data will be aligned at the SPE octet and occupy the full forty-eight octets for the OC48*c* frame.

[0198] The InfiniBand end nodes **62***a* interface to the OIR system **31***c* through the InfiniBand interface. The InfiniBand frames are encapsulated into the OIR Point-to-Point packet **140**. The packet is then mapped into the SONET SPE and forwarded to the destined OIR system **31***c*. At the destined OIR system, the OIR system will strip out the InfiniBand frames from the OIR packet before forwarding it to the InfiniBand end nodes **62***a*.

[0199] **FIG. 10** illustrates the method of how the InfiniBand packets are switched using the DWDM Interfaces **67**. The DWDM interface is a more effectively way of transporting data between optical system. It is a fiber-optic transmission technique that involves the process of multiplexing a multitude of wavelength signals onto a single fiber. In the OIR system **31***d*, each DWDM Interface card **43** can support a plurality of wavelength signals on each port. The DWDM layer within the OIR system has been designed in compliance with industry standards (see reference [13]). The bit rate and protocol transparency allows the DWDM interface to transport native enterprise data traffic like InfiniBand, Gigabit Ethernet, Fibre Channel, SONET, IP, iSCSI, etc. on different channels. It brings the flexibility to the OIR system in relation to the overall transport system; it can connect directly to any signal format without extra equipment.

[0200] The OIR system contains an optical amplifier that is fueled by a compound called Erbium, operated in a specific band of the frequency spectrum. It is optimized for interfacing with existing fiber and can carry a multitude of lightwave channels.

[0201] InfiniBand frames transported over DWDM use Point-to-Point (PPP) protocol. PPP protocol uses the DWDM transport as a byte oriented full-duplex link. The OIR system will use the lightweight SONET layer approach to transport OIR Packet over the DWDM transport. That is, the OIR system will preserve the SONET header as a means of framing the data but will not use the Time Division Multiplexing (TDM) approach to transport payload. The OIR packet is transported to the next OIR system **31***d* "as is". The OIR system **31***d* will have the intelligence to add and drop wavelengths at the destination OIR system **31***d*.

[0202] Forward Error Correction (FEC) function is performed in all OIR systems **31***d* to provide the capability to detect signal errors. The FEC data is put into the unused portion of the SONET header. Network restoration and survivability functions will be supported by the Multiple Protocol Lambda Switching (MPLS) protocol (see reference [11]).

[0203] OIR systems **31***d* can interconnect to the InfiniBand end nodes **62***a* by establishing a light path between the two end nodes. This light path is a logical path that is established so that the optical signal can traverse the intermediate OIR system **31***d* to reach the destination end node from an originating end node.

[0204] The InfiniBand end nodes **62***a* interface to the OIR system **31***d* through InfiniBand interfaces **63**. The InfiniBand frames are encapsulated into the OIR Point-to-Point packet **140**. Based on the destination address, a route and wavelength are assigned to carry the OIR packet. The packet is then inserted into the wavelength transport and forwarded to the destination OIR system **94, 95**. At the destination OIR system, the Optical-Electrical-Optical (OEO) function is performed to convert the OIR packet into machine-readable form. The OIR system **31***d* will then strip out the InfiniBand frames **150** from the OIR packet **140** before forwarding it to the InfiniBand end nodes **62***a*.

[0205] **FIG. 11** illustrates the method of how the Fibre Channel Frames are switched using the DWDM Interfaces **67**. The operation in transporting the Fibre Channel frames through the DWDM interface of the OIR system network is similar to what has been discussed in previous paragraphs.

[0206] The Fibre Channel end nodes **62***b* interface to the OIR system **31***d* through Fibre Channel interfaces **68**. The Fibre Channel frames are encapsulated into the OIR Point-to-Point packet **140**. Based on the destination address, a route and wavelength are assigned to carry the OIR packet. The packet is then inserted into the wavelength transport and forwarded to the destination OIR system **31***d*. At the destined OIR system **31***d*, the Optical-Electrical-Optical (OEO) function is performed to convert the OIR packet into machine-readable form. The OIR system will then strip out the Fibre Channel frames **160** from the OIR packet **140** before forwarding it to the Fibre Channel end nodes **62***b*.

[0207] **FIG. 12** illustrates the method of how the InfiniBand Host Client can interface with the Fiber Channel Target device through the OIR system InfiniBand/Fibre Channel Gateway function. The InfiniBand Frames switching between OIR system **31***d* is the same as described in discussion for **FIG. 10**. The major difference is that the destination OIR system**31***d* will perform the InfiniBand/Fibre Channel gateway function to bridge the InfiniBand data and the Fibre Channel data.

[0208] To support the InfiniBand/Fibre Channel gateway function, the user will provision and activate the InfiniBand/Fibre Channel Gateway 121 function at the OIR system **31***d*. A gateway server function **121** will be started and it will also setup the link between the Fibre Channel devices that are connected to the OIR Fibre Channel Interface ports **68**. The gateway server will automatically setup the links with the Fibre Channel devices.

[0209] The gateway server will also advertise itself to the other InfiniBand Subnet Management Agents (SMA) (as described in InfiniBand Architecture Specification, reference [1]) about the existence of InfiniBand target devices. The InfiniBand end node **62***a*, which is acting as a Host Server, will treat the Fibre Channel devices attached to the OIR system **31***d* as targets; it will be able to perform InfiniBand operations on them.

[0210] The InfiniBand data are carried from the Client 61*a*, through the intermediate OIR system **31***d* to the destination OIR system **31***d*. The InfiniBand frame data **150** is stripped from the OIR packet **140** and is forwarded to the InfiniBand/Fibre Channel gateway server **121**. The gateway server **121** converts the InfiniBand data **150** into meaningful Fibre Channel commands/control information **160** and passes it down to the Fibre Channel device **62***b* through the destination Fibre Channel Interface port **68**. The Fibre Channel device **62***b* that is attached to the Fibre Channel Interface port **68** will respond to the Fibre Channel commands/control information **160** as required. A similar process is performed when the Fibre Channel device **62***b* returns the storage data to the InfiniBand host **62***a*.

[0211] **FIG. 13** illustrates the method of how the InfiniBand Host Client 61*a* can interface with the iSCSI Target device **62***c* through the OIR system InfiniBand/iSCSI Gateway function **131**. The InfiniBand Frames switching between OIR systems **31***d* is the same as described in discussion for **FIG. 10**. The major difference is that the destination OIR system will perform the InfiniBand/iSCSI gateway function to bridge the InfiniBand data **150** and the iSCSI data **180**.

[0212] iSCSI is a storage networking technology, which allows users to use high-speed SCSI (Small Computer Systems Interfaces) devices through out Ethernet networks. Natively, the OIR system **31***d* allows SCSI data to be transported through the OIR system **31** network using the Gigabit Ethernet interfaces **64**. However, when InfiniBand is used from the Client **61***a* to access iSCSI devices **62***c*, the OIR system **31***d* can provide an additional benefit.

[0213] The benefit of using the OIR system **31** is that the Client **61***a* can perform the same InfiniBand operation 61*b* on a plurality of devices, including InfiniBand Target devices **62***a*, Fibre Channel devices **62***b*, and iSCSI devices **62***c*. Similar to the discussion on InfiniBand/Fibre Channel gateway operation, the InfiniBand data **150** will be converted to ISCSI command/control information **180** by the InfiniBand/iSCSI Gateway server **131**. The iSCSI information **180** is forwarded by the OIR system **31***d* through its Gigabit Ethernet interface **64** to the iSCSI device **62***c*.

Data Format—FIG. 14, 15, 16, 17, and 18

[0214] **FIG. 14** illustrates the Optical InfiniBand Router (OIR) Point-to-Point packet format **140**. The OIR packet

140 is based on a HDLC-like Point-to-Point framing format described in IETF RFC 1662 (see references [2]; and [3]). The following describes the field information:

[0215] Flag **141, 148**—The Flag Sequence indicates the beginning or end of a frame.

[0216] Address **142**—The Address field contains the binary sequence 11111111, which indicates "all station address". PPP does not assign individual station addresses.

[0217] Control **143**—The Control field contains the binary sequence 00000011.

[0218] Protocol ID **144**—The Protocol ID identifies the network-layer protocol of specific packets. The proposed value for this field for InfiniBand is 0×0042, Fibre Channel is 0×0041, and iSCSI is 0×0043. (Internet Protocol field value is 0×0021).

[0219] Label **145**—The Label field supports the OIR Label switching function.

[0220] Information field **146**—Data frame is inserted in the Information field with a maximum length of 64 K octets. (Note: the default length of 1,500 bytes is used for small packet).

[0221] FCS (Frame Check Sequence) field **147**—A 32-bit (4 bytes) field provides the frame checking function. (Note: 32 bits instead of 16 bits is used to improve error detection.)

[0222] **FIG. 15** illustrates the method of how an InfiniBand Frame **150** is encapsulated within the Optical InfiniBand Router (OIR) Point-to-Point packet format The following describes the field information for the InfiniBand Frame:

[0223] Routing Header **150*a*** —contains the fields for routing the packet between subnets.

[0224] Transport Header **150*b*** —contains the fields for InfiniBand transports.

[0225] Payload **150*c*** —contains actual frame data.

[0226] CRC **150*d*** —Cyclic Redundancy Check data

[0227] **FIG. 16** illustrates the method of how a Fibre Channel Frame **160** is encapsulated within the Optical InfiniBand Router (OIR) Point-to-Point packet format **140**. The following describes the field information for the Fibre Channel Frame:

[0228] Start of Frame **160*a*** —indicates beginning of a frame.

[0229] Fibre Channel Header **160*b***—contains control and addressing information associated with the Fibre Channel frame .

[0230] Optional Header **160*c***—contains a set of architected extensions to the frame header.

[0231] Payload **160*d***—contains actual frame data.

[0232] CRC **160*e*** —Cyclic Redundancy Check data

[0233] End of Frame **160*f***—indicates end of a frame

[0234] **FIG. 17** illustrates the method of how an Ethernet Frame **170** is encapsulated within the Optical InfiniBand Router (OIR) Point-to-Point packet format **140**. The following describes the field information for the Ethernet Frame **170**:

[0235] Preamble **170*a*** —indicates beginning of a frame. The alternating "1, 0" pattern in the preamble is used by the Manchester encoder/decoder to "lock on" to the incoming receive bit stream and allow data decoding.

[0236] Start Frame Delimiter (SFD) **170*b*** —is defined as a byte with the "10101011" pattern.

[0237] Destination Address (DA) **170*c*** —denotes the MAC address of the receiving node.

[0238] Source Address (SA) **170*d*** —denotes the MAC address of the sending node.

[0239] Length (LEN) **170*e*** —indicates the frame size.

[0240] Data **170*f*** —contains actual frame data.

[0241] PAD **170*g*** —contains optional padding bytes.

[0242] Frame Check Sequence (FCS) **170*h*** —for error detection.

[0243] **FIG. 18** illustrates the method of how iSCSI Frame **180** is encapsulated within the Optical InfiniBand Router (OIR) Point-to-Point packet format **140**. The iSCSI Frame **180** is basically SCSI data encapsulated within the IP Packet, which in turn is wrapped within the Ethernet frame **170**. The following describes the Internet Protocol (IP) field information:

[0244] IP Header **181**—contains the Internet Protocol Header Information.

[0245] SCSI **182**—contains SCSI commands.

[0246] **FIG. 19** illustrates the method of how InfiniBand Processing System **55** processes the input data, while **FIG. 20** illustrates the method of how the said InfiniBand Processing System **55** processes the output data.

[0247] **FIG. 21** illustrates the method of how Gigabit Ether-Channel Processing System **51** processes the input data, while **FIG. 22** illustrates the method of how the said Gigabit Ether-Channel Processing System **51** processes the output data.

[0248] **FIG. 23** illustrates the method of how Fibre Channel Processing System **56** processes the input data, while **FIG. 24** illustrates the method of how the said Fibre Channel Processing System **56** processes the output data.

[0249] **FIG. 25** illustrates the method of how Processing Systems for OC-48 SONET interface, OC-192 SONET interface, DWDM interface, and 10-Gigabit Ethernet interface **53, 57, 54, 52** process the input data, while **FIG. 26** illustrates the method of how the said Processing Systems **53, 57, 54, 52** process the output data.

CONCLUSION, RAMIFICATIONS, AND SCOPE

[0250] In addition to the combined InfiniBand switching and routing functions, the OIR system provides system and network multi-services for the following areas:

[0251] InfiniBand packets over Gigabit Ethernet Channels (Ether-Channel) for inter-subnet routing

[0252] InfiniBand packets over Ether-Channels and SONET for inter-network routing

[0253] InfiniBand packets over Multi-Wavelength DWDM for WAN-based inter-domain routing/transport

[0254] InfiniBand packets to Storage Area Network gateway (Fibre Channel gateway) function

[0255] InfiniBand packets to Network Attached Storage gateway (iSCSI gateway) function

[0256] Full InfiniBand Network Domain Management

[0257] InfiniBand Quality of Service (QoS)/Bandwidth control to Optical Network QoS/Bandwidth control mapping functions

[0258] This invention takes advantages of the InfiniBand architecture, extending it to incorporate the InfiniBand capabilities to go beyond the local area network. By using the optical networking capabilities, it allows processing modules and I/O modules to be connected through the local network, through the metro area network, and even to the wide area network.

[0259] In addition to the multi-services support functions, the OIR also include the following features to provide a highly reliable infrastructure:

[0260] Fully NEBS-compliant hardware platform

[0261] Interchangeable line card modules

[0262] Non-blocking, redundant switching fabric ensures highest service quality

[0263] Support for multiple access and transport types, including InfiniBand, Gigabit Ethernet, SONET, DWDM

[0264] Full 1+1 redundancy protects management processors and switching fabric modules

[0265] Hot-swappable components and support for online software and firmware upgrades offer the highest availability

[0266] Remote management tools accommodate either conventional or next generation network management systems

[0267] Replaces multiple network elements by performing functions that include InfiniBand switching and routing, IP switching and routing, SAN/NAS gateway functions, and SONET/DWDM payload switching

[0268] This invention will be unique and easily differentiated from competitive products because of its comprehensive service management solution, including network, system, and application levels management. It offers the simplicity of Ethernet technology, combined with the reliability and performance of the optical technology. It allows the customers to tune the system to deliver scalable, guaranteed rate access to multiple network services. This will give our customer the important time-to-market and differentiated service advantage they need to compete in the new networking market.

[0269] To the potential customer, the OIR is the natural choice given its multi-service nature, speed, and undisputed cost advantage. OIR also brings new dimensions of simplicity compare to earlier generation wide-area network (WAN) access technologies. It will become the service demarcation point for traffic in LAN, SAN, NAS, MAN, and WAN.

[0270] Multi-service access eliminates the incorporation of multiple networking transport switches/routers within a data center. Any service can be attached to the OIR without the complexity in managing the different characteristics of multi-vendor equipment.

[0271] Traffic is encapsulated into the OIR transport and groomed to high-speed SONET/SDH paths, or trunks, which ultimately terminates at the required Internet, native Ethernet, and/or InfiniBand-based service destination. Efficiency is assured with advanced bandwidth management capabilities plus the ability to share "trunks" among multiple customers and across multiple platforms

[0272] This invention simplifies the overall system network architecture by collapsing the capabilities of InfiniBand, IP switches and routers, SONET Add-Drop Multiplexers, and DWDM into one cost-effective and powerful optical router. Potential customers can select one or more service components that they want to use within our system. The service components can be interfaces for InfiniBand (2.5 gigabit or 10 gigabit), Gigabit Ethernet (3×1 gigabit or 10 gigabit), SONET (OC-48 or OC-192), or DWDM (4 channels OC-48 or 4 channels OC-192).

BEST MODE FOR CARRYING OUT THE INVENTION

[0273] The problems solved by this invention is:

[0274] how to extend the System-Area Networking of the InfiniBand technology beyond the limited distance. The current specification defines the fiber connection distance to be less than 100 meters.

[0275] how to transport and route data between InfiniBand devices using the Gigabit Ethernet-based data transport.

[0276] how to combine a plurality of Gigabit Ethernet data streams into one InfiniBand data stream.

[0277] how to segment data between InfiniBand devices and the Gigabit Ethernet-based devices

[0278] how to transport and route data between InfiniBand devices using the SONET Add-Drop Multiplexing data transport.

[0279] how to transport and route data between InfiniBand devices using the Dense Wavelength Division Multiplexing (DWDM) data transport.

[0280] how to transport and route data between Fibre Channel devices using the Dense Wavelength Division Multiplexing (DWDM) data transport.

[0281] Operationally, one uses the Optical InfiniBand routing device to transport data from InfiniBand host or target devices through the OIR network to the destination InfiniBand host or target devices.

[0282] One can also use the OIR routing device to transport IP data, Fibre Channel data, or SCSI data through the OIR device to the destination devices. The OIR device has the capabilities to encapsulate any data and transport or route them to destinations that are supported by the OIR device.

[0283] When one uses the Gigabit Ethernet interface as the backbone transport, data such as InfiniBand, IP, Fibre Channel, and SCSI, are encapsulated into an OIR generic packet and passed down to the Gigabit Ethernet Media Access Layer (MAC) for data transport. When the data packet arrives at the destination, the data packet is stripped out from the Gigabit Ethernet Frame. The data packet header is inspected to determine the processing required. The raw data will be stripped from the data packet and forwarded to the destination interface.

[0284] Similar processing is done when one uses the SONET interface as the backbone transport, data such as InfiniBand, IP, Fibre Channel, and SCSI, are encapsulated into an OIR generic packet and passed down to the SONET framing processor for data transport. When the data packet arrives at the destination, the data packet is stripped out from the SONET Frame. The data packet header is inspected to determine the processing required. The raw data will be stripped from the data packet and forwarded to the destination interface.

[0285] When one uses the DWDM interface as the backbone transport, data such as InfiniBand, IP, Fibre Channel, and SCSI, are encapsulated into an OIR generic packet and passed down to the DWDM processor for data transport. When the data packet arrives at the destination, the data packet is stripped out from the DWDM payload. The data packet header is inspected to determine the processing required. The raw data will be stripped from the data packet and forwarded to the destination interface.

ADVANTAGES OVER THE PRIOR ART

[0286] Accordingly, besides the objects and advantages of supporting multiple networking/system services described in my above patent, several objects and advantages of the present invention are:

[0287] to provide a system which can extend the transport of InfiniBand from the 100-meter limit to beyond 100 K meters

[0288] to provide a system which can transport InfiniBand data through Gigabit Ethernet interface between the InfiniBand host or target channel devices.

[0289] to provide a system which can transport InfiniBand data through the SONET Add-Drop Multiplexer interface between the InfiniBand host or target channel devices.

[0290] to provide a system which can transport InfiniBand data through the DWDM interface between the InfiniBand host or target channel devices.

[0291] to provide a system which can provide a gateway function, which can transport InfiniBand data streams to/from Network Attached Storage Filer devices.

[0292] to provide a system which can provide Quality of Service control over the InfiniBand data streams through the OIR network. The OIR network can be comprised of Gigabit Ethernet interfaces, SONET interfaces, Fibre Channel interfaces and DWDM interfaces.

[0293] Further objects and advantages are to provide a highly reliable, highly available, and highly scalable system, which can be upgradeable to different transport services, including Gigabit Ethernet, SONET, and DWDM. The system is simple to use and inexpensive to manufacture compared to the current Gigabit Ethernet-based IP routers, SONET Add-Drop Multiplexers, and DWDM devices. Still further objects and advantages will become apparent from a consideration of the ensuing description and drawings.

OPERATION OF INVENTION

[0294] The manner in which the OIR system will be used is as follows:

[0295] to connect the InfiniBand Target Channel Adapter (TCA) optical cables or Host Channel Adapter (HCA) optical cables to the OIR InfiniBand optical port on an InfiniBand interface card. A plurality of TCA and HCA can be connected to the OIR InfiniBand optical port. In addition, a plurality of OIR InfiniBand interface card can be added to support additional connections. Upon connection, InfiniBand data streams can be transferred between the TCA and HCA devices.

[0296] to connect the Gigabit Ethernet (GE) optical cables to the OIR InfiniBand optical port on Gigabit Ethernet interface card. A plurality of Gigabit Ethernet networking devices can be connected to the OIR InfiniBand optical port. In addition, a plurality of OIR GE interface card can be added to support additional connections. Upon connection, Ethernet data streams can be transferred between the Ethernet devices. Currently, Gigabit Ethernet networking devices, other than the OIR system, carries only IP packets. In this situation, the OIR system will act as a high-speed IP router.

[0297] to connect the Gigabit Ethernet (GE) optical cables to the OIR InfiniBand optical port on Gigabit Ethernet interface card. A plurality of OIR systems can be connected to the OIR GE optical port. In addition, a plurality of OIR GE interface card can be added to support additional connections. Upon connection, OIR data packets can be transferred between the OIR systems. In this situation, the OIR system will act as a high-speed router for a plurality of data traffic, including InfiniBand, IP, Fibre Channel, and SCSI.

[0298] to connect the SONET optical cables to the OIR InfiniBand optical port on SONET interface card. A plurality of OIR systems or SONET Add-Drop Multiplexers can be connected to the OIR SONET optical port. In addition, a plurality of OIR SONET interface card can be added to support additional connections. Upon connection, OIR data packets can be transferred between the OIR system and SONET Add-Drop Multiplexing devices. In this

situation, the OIR system will act as a high-speed SONET transporter for a plurality of data traffic, including InfiniBand, IP, Fibre Channel, and SCSI.

[0299] to connect the DWDM optical cables to the OIR DWDM optical port on DWDM interface card. A plurality of OIR systems or DWDM can be connected to the OIR SONET optical port. In addition, a plurality of OIR SONET interface cards can be added to support additional connections. Upon connection, OIR data packets can be transferred between the OIR system and DWDM devices. In this situation, the OIR system will act as a high-speed DWDM transporter for a plurality of data traffic, including InfiniBand, IP, Fibre Channel, and SCSI.

I claim:

1] A system comprises of a plurality of network interface devices, having the capabilities to route data from one network interface device to a plurality of network interface devices within the same said system, wherein the said system comprises:

A plurality of management devices;

A plurality of switching fabric devices;

A plurality of network interface devices that can encapsulate respective network interface protocol data into a common data packet that is used to route amongst the network interface devices within the said system;

Route means for forwarding a data packet from the source network device to destination network device; or from the source network device to a destination intermediate said system within a networked environment.

2] A system according to claim **1**, wherein the source network device is an InfiniBand device, the data sent to the said optical device is InfiniBand frames, and the said system can forward the InfiniBand frames to the destination network device that is a InfiniBand device.

3] A system according to claim **1**, wherein the source network device is a Fiber Channel device, the data sent to the said optical device is Fibre Channel frames, and the said system can forward the Fibre Channel frames to the destined network device that is a Fibre channel device.

4] A system according to claim **1**, wherein the source network device is a Gigabit Ethernet device, the data sent to the said optical device is Ethernet frames, and the said system can forward the Ethernet frames to the destined network device that is an Ethernet device.

5] A system according to claim **1**, wherein the source network device is an InfiniBand device, the data sent to the said optical device is InfiniBand frames, and the said system can forward the InfiniBand frames to the destination network device that is a Fibre Channel device.

6] A system according to claim **1**, wherein the source network device is a Gigabit Ethernet device using IP protocol, the data sent to the said optical device is SCSI command encapsulated within IP packets, and the said system can forward the IP packet to the destination network device that is a iSCSI device.

7] A plurality of said system according to claim **1** connected together to form a system network, wherein the network interface used by the said system within the said network is InfiniBand; and wherein the said system can route the data according to claim **2** from the source network device to the destined network device through the said system network.

8] A plurality of said system according to claim **1** connected together to form a system network, wherein the network interface used by the said system within the said network is Gigabit Ethernet; and wherein the said system can route the data according to claim **2**, claim **3**, claim **4**, claim **5** and claim **6** from the source network device to the destination network device through the said system network.

9] A plurality of said system according to claim **1** connected together to form a system network, wherein the network interface used by the said system within the said network is SONET; and wherein the said system can route the data according to claim **2**, claim **3**, claim **4**, claim **5** and claim **6** from the source network device to the destination network device through the said system network.

10] A plurality of said system according to claim **1** connected together to form a system network, wherein the network interface used by the said system within the said network is DWDM; and wherein the said system can route the data according to claim **2**, claim **3**, claim **4**, claim **5** and claim **6** from the source network device to the destination network device through the said system network.

* * * * *